# ソフトウェア AUI ユーザビリティガイドライン

著者：諸熊浩人

2012 年 4 月 3 日

株式会社 U'eyes Design

# ソフトウェア AUI ユーザビリティガイドライン

諸熊　浩人

株式会社 U'eyes design

2012 年 4 月 4 日

---

## 目次

記号の説明 ........ 3
序章 ........ 4
第１部　AUI ユーザーを学ぶ ........ 8
　第１章『AUI ユーザー』 ........ 9
　第２章『AUI と GUI の違い』 ........ 15
第２部　基本的 AUI ユーザビリティを学ぶ ........ 20
　第３章『ラベリング』 ........ 21
　第４章『画面デザイン』 ........ 26
　第５章『操作性』 ........ 32
　第６章『音』 ........ 40
　第７章『ドキュメント』 ........ 44
第３部　より深く AUI ユーザビリティを学ぶ ........ 53
　第８章『音声化される情報をコントロールする』 ........ 54
　第９章『音声読み上げを制御する』 ........ 58
　第 10 章『AUI10 の原則』 ........ 67
参考文献 ........ 73
付録資料 ........ 74
著作情報 ........ 93

---

## 記号の説明

本書に登場する記号には、以下のような意味があります。

| 表１『記号の使い分け』 | |
|---|---|
| 記号名 | 説明 |
| (・・・) | 用語の別名、または短い説明として用います。 |
| 「・・・」 | 会話、または読み上げる内容などに用います。 |
| 『・・・』 | 製品・著書名に用います。 |
| 【 コラム 】 | 複数行のコラムに用います。 |
| ◆・・・ | 項目名(小見出し)として用います。 |
| ※・・・ | 一行補足として用います。 |
| 下線付き文字 | ハイパーリンクになっている文字に表示されることがあります。 |

# 序章

## ◆はじめに

現在、私達の生活を支えているパソコンには、さまざまなアプリケーションソフトウェアが搭載されています。その種類は豊富で、ワープロ・表計算・電子メールといった現代の必需品となっているものから、音楽・アート・ゲームといった娯楽、さらには日々のスケジュールの管理や、会計・経理を行なうアプリケーションまで多岐に渡ります。

しかし、全てのユーザがその恩恵を受けられているわけではありません。例えば視覚障害があり画面の表示が見えないユーザは、現代のグラフィック中心のアプリケーションソフトウェアをほとんど利用することはできません。

著者もこの視覚障害ユーザーとなった一人です。視力を失ったことで、それまで自然に行なえたはずのパソコン操作の全てが不可能になりました。

しかし、視覚障害ユーザーでもそれらのアプリケーションソフトウェアが使えるようになる方法があります。それは、操作するために必要となる情報を、音声や効果音を用いてフィードバックすることです。

このユーザインタフェース **AUI(Auditory User Interface)**は、著者を初めとする多くの視覚障害ユーザーに普及しました。少なくともその結果、著者は音の情報を頼りに本書を執筆できたのです。

とはいえ、まだ視覚障害ユーザーが全てのアプリケーションソフトウェアを使うことはできません。現代の GUI(Graphical User Interface)ユーザーを規準に制作したソフトウェアの全てを音声や効果音で表現することは完全ではありません。

また、音声や効果音を頼りに操作する方法や、その時の考え方についても、まだ充分に確立されているわけではありません。そこで本書では、この視覚障害ユーザーと、ユーザインタフェースにスポットライトを当てることにしました。

## ◆AUI

**AUI** とは、コンピュータを操作するためのユーザインタフェースの一つです。

AUI の特長は、ユーザがマウスやキーボードを用いて行なった操作の結果や、現在のパソコンがどんな状態であるかなどを、音声や効果音でフィードバックすることです。

このユーザインタフェースを最も簡単に実現できるツールは、**スクリーンリーダー**と呼ばれるアプリケーションソフトウェアです。

スクリーンリーダーとは、現在多くの視覚障害のユーザに利用されている支援ソフトウェアです。画面に表示された情報や、ユーザが操作した結果を音声で読み上げることができます。

**◆ガイドラインの概要**

本書は、AUI を利用するユーザが使いやすいソフトウェアを開発するためのガイドラインです。

ユーザビリティとは、既存の製品やサービスを、ある限定された条件下で使った時、その利用目的に対する達成度合いを総称したものです。より詳しい定義については、ISO 9241-210『Ergonomics of human-system interaction -- Part 210: Human-centred design for interactive systems』(日本版 JIS Z8530『人間工学―インタラクティブシステムの人間中心設計プロセス』)を参照して下さい。

このガイドラインで目標としていることは、AUI ユーザーが、正確に作業を遂行できること、作業時間を短縮できること、そして、利用する過程で感じるストレスを軽減できることを念頭に置いたアプリケーションソフトウェアを開発することです。

また同時に、本書は AUI ユーザーへ向けての**アクセシビリティ**に配慮してソフトウェアを設計するためのガイドラインでもあります。

**アクセシビリティ**とは、「ユーザが指定した機能にアクセスできるようになっているかどうか」を意味する言葉で、前述の**ユーザビリティ**の**正確に遂行できること**が達成されるために必須となる要素です。

本書では主に第２部で、AUI ユーザーが全ての機能にアクセスし操作できるアプリケーションソフトウェアを開発するための要点を記述しています。

**◆ガイドラインの特長**

このガイドラインには、次のような特長があります。

- AUI ユーザーを**知る**セクションを設けているので、AUI ユーザーにとってわかりやすいソフトウェアが制作できます。
- 音声や効果音を頼りに情報を素早く検索できるための用法が盛り込まれているので、目的の作業を素早く遂行できるソフトウェアが制作できます。
- 操作ミスや混乱を防ぐと共に、誤った際の対処法についても紹介しているので、アプリケーションを使う上でのストレスを軽減できるようなソフトウェアが制作できます。

またこのソフトウェアは、AUI ユーザー以外のユーザにとっても、以下のように使いやすくなる特長があります。

- 画面を見続けなくても使用できるソフトウェアになるので、注視することによる疲労を少なくできます。
- 全ての機能をキーボードでも操作できるようになるので、マウス操作の苦手なユーザや、ショートカットキーを駆使して素早く操作したいユーザのニーズにも対応できます。
- 視力が低いために画面を拡大しているユーザや、小さなディスプレイでパソコンを利用しているユーザにとって、通常の GUI アプリケーションよりも使いやすくなります。

本書は、他のユーザビリティガイドラインと比較すると、以下のような特長を持っています。

- 障害当事者自身が制作したガイドラインなので、視覚障害ユーザーの視点を盛り込んでいます。
- 著者は 5 年以上に渡りソフトウェア制作の経験を積んでおり、現在もアプリケーション制作を行なっています。従って、制作に実装できる用法も説明に加えています。

### ◆本書の構成

本書は、3 つの部で構成しています。

第 1 部　AUI ユーザーを学ぶ

AUI を利用している視覚障害のユーザの特長や課題、AUI とはどういった特長があるのかを紹介します。

この部には、具体的なソフトウェア開発に活用できるガイドラインはありません。しかし、AUI ユーザーを知ることで、第 2 部以降のガイドラインがわかりやすくなりますので、ぜひ目を通して下さい。

第 2 部　AUI アクセシビリティを学ぶ

AUI ユーザビリティの基本として、**アプリケーションの全ての操作を AUI で実現できる**ことを念頭に置いた部です。そのために、アプリケーション開発時に配慮して欲しい要点を紹介しています。

この部で挙げている多くの記述は、ユーザビリティの仲でも、**全ての機能にアクセスできる**という意味を持つ用語**アクセシビリティ**に関する内容です。

第 3 部　より深く AUI ユーザビリティを学ぶ

AUI の中核である音声情報の取り扱い方について紹介します。第 2 部のガイドラインを理解した上で、より優れたユーザビリティを追求したい時に活用して下さい。

この部では、どのように音声情報を提示すれば AUI ユーザーに使いやすいアプリケーションになるのか、その要点を中心に記述しています。

### ◆対象とする読者

本ガイドラインは、次のような方には特に読んでいただきたいです。

- アプリケーション開発やコンテンツ制作に携わっている方
- 障害者や障害者支援に関心のある方
- 音や音声のユーザインタフェースに関心のある方

ただし、本書には以下の内容については充分には記述していません。

- 音声入力や、点字・触覚に関わるユーザインタフェース
- 具体的なソースコードの書き方や、ソフトウェアの名称
- Windows 向けアプリケーションソフトウェア以外の製品へのアクセシビリティの応用

※本書を Windows 向けアプリケーションに限定して記述している理由は、日本では Windows 向けスクリーンリーダーの入手が最も容易であるからです。

## 第１部　AUI ユーザーを学ぶ

第１部では、AUI ユーザビリティを発揮できるアプリケーションソフトウェアを開発するにあたって、まず AUI ユーザーについて紹介します。
優れたユーザビリティを発揮できる製品を開発するための、最も重要な方法は、対象としているユーザを知ることです。その理由は、対象とするユーザは、開発者と異なる環境、常識、そして経験を持っているからです。

例えば、AUI でアプリケーションソフトウェアを操作すると、何か一つ操作を行なう度に、その結果は音声や効果音で通知されます。ただし、ユーザは、読み上げた内容を聞いて**何が起こったのか**がわかったら、読み上げの音声を停止させ、次の操作を行ないます。
このため健常の人が、視覚障害ユーザーがスクリーンリーダーで読み上げている音声だけを聞くと、音声が途中で頻繁に途切れていて、いったい何をやっているのかを予測することはできません。そう思う理由は、音声読み上げが速いからではなく、音声に対する付き合い方が違うからです。
『スクリーンリーダーで操作している音声』を録音しました。音声の前半では、前の段落のテキストを入力しており、後半では、入力した内容を読み上げさせて確認しています。

## 第１章『AUI ユーザー』

本書における AUI ユーザーとは、視覚障害ユーザーのことです。特にパソコンの画面を見ることが困難で、スクリーンリーダーによる音声読み上げを利用しているユーザのことを指します。

この章では、AUI ユーザーがどのようにパソコンを利用しているのかについて紹介します。

---

### ◆AUI ユーザーのパソコン操作

AUI ユーザーはパソコンに**スクリーンリーダー**という画面読み上げソフトウェアを導入した上で、**キーボード**を用いて操作します。

多くの GUI ユーザーがパソコンを使う時には普通、ディスプレイに表示された情報は、それを**見て**確認します。しかし、AUI ユーザーは**見る**ことができないため、代替手段が必要となります。

そこで用いられているのが**音**です。AUI ユーザーは、ディスプレイに表示される文字や、現在の操作の状況を音声読み上げや効果音を用いて入手し、それをヒントにして操作を行ないます。

また AUI ユーザーはマウスを使わず、キーボードだけでパソコンを操作します。なぜならば、マウスで操作しようとしても、マウスカーソルが今、画面のどの位置にあって、どの位置へ移動させれば良く、そしてマウスを動かした際にカーソルがどのくらい移動したのかを知ることができないからです。

このことから、AUI ユーザーがアプリケーションソフトウェアを利用できるためには、**全ての機能**を**キーボード**だけでも操作でき、且つその一連の操作の方法や結果を、**スクリーンリーダー**が読み上げなければなりません。

### ◆スクリーンリーダー

スクリーンリーダーとは、ディスプレイに表示された内容や、ユーザが操作した結果などを合成音声で読み上げるアプリケーションソフトウェアです。

スクリーンリーダーを導入すると、どんなパソコンでも音声読み上げ機能を利用することができます。AUI の利用や検証を検討している時や、画面の表示が見えにくいと感じている時、またはパソコンをしゃべらせてみたい時には、ぜひ試してみることをお薦めします。

現在、多くの AUI ユーザーが使用しているスクリーンリーダーの紹介・入手先についての詳細は、付録１『スクリーンリーダー情報』を参照して下さい。

以下では、スクリーンリーダーが、どのようにパソコンの画面を音声で読み上げているのかについて紹介します。

下の画像では、まず現在前面に表示されているウィンドウの名称を「デスクトップ」と読み上げます。続いて、現在のウィンドウには複数の項目があり、いずれかを選択することができることを「リストビュー」と読み上げます。

画像１『デスクトップの画面』

ユーザがキーボードの矢印キーを押すと、画面に表示されているアイコン間を移動することができ、移動した先にあるアイコンの名称を読み上げます。上の画像では「スクリーンリーダー」のアイコンが選択されているので、そのアイコン名である「スクリーンリーダー」と読み上げます。

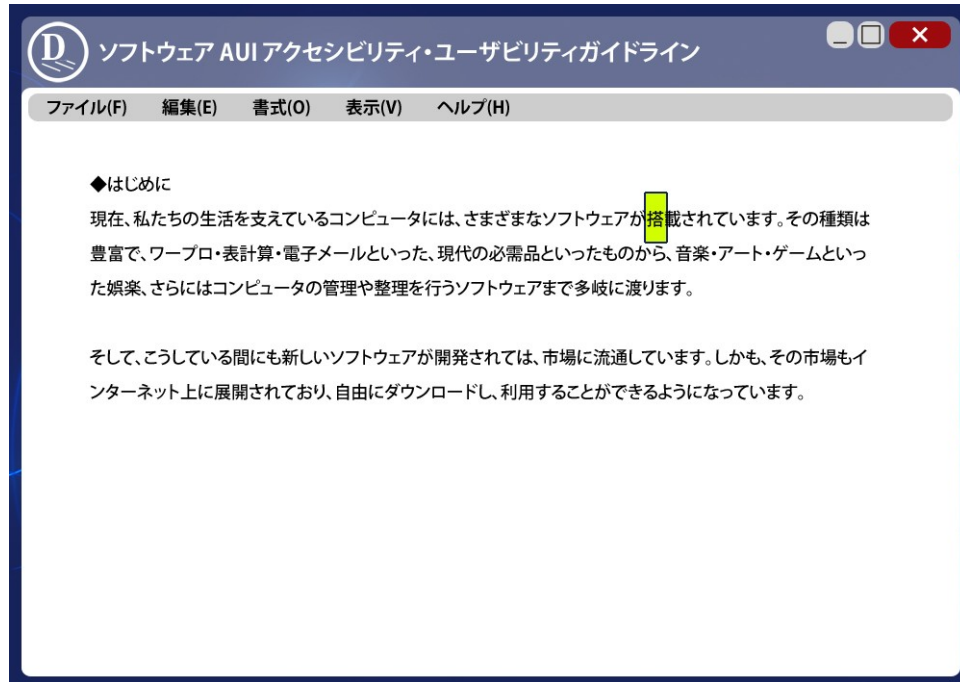
この時、すでに下端の項目を選択しているのにさらに下へカーソルを移動させようとすると、「下端　スクリーンリーダー」と読み上げます。これによって、ユーザは**これ以上はカーソルが移動しないので、項目が無い**ことを知ることができます。

次の事例は、テキストエディタで文字を入力している画面です。ここでも、まず最初に、読み込んでいるファイル名、続いてアプリケーション名を読み上げます。下の画面の場合は、「ソフトウェア　エーユーアイユーザビリティガイドライン　マイナス　メモチョー」と読み上げます。

さらにこの画面では、**文字を入力することができる**ことを通知します。具体的には、**複数行に渡って文字が入力できる**ことを意味する「エディットリョーイキ」という内容を付け加えてくれます。

この「エディットリョーイキ」という読み上げ内容は、文字を入力できるウィンドウであれば、他のアプリケーションでも同じように聞くことができます。例えば電子メールを相手に送るために本文を書く画面でも、「ホンブン　エディットリョーイキ」と読み上げます。

画像 2『「搭」を選択している画面』

ソフトウェア AUI アクセシビリティ・ユーザビリティガイドライン
ファイル(F)　編集(E)　書式(O)　表示(V)　ヘルプ(H)
◆はじめに
現在、私たちの生活を支えているコンピュータには、さまざまなソフトウェアが搭載されています。その種類は豊富で、ワープロ・表計算・電子メールといった、現代の必需品といったものから、音楽・アート・ゲームといった娯楽、さらにはコンピュータの管理や整理を行うソフトウェアまで多岐に渡ります。
そして、こうしている間にも新しいソフトウェアが開発されては、市場に流通しています。しかも、その市場もインターネット上に展開されており、自由にダウンロードし、利用することができるようになっています。

このような画面では、上下の矢印キーを押してカーソルを上下に移動させると、その行の先頭から末尾までの文字を一度に読み上げます。何も書かれていない行では「空行」と読み上げます。

また左右の矢印キーを押して一文字を選択すると、その文字の詳細を読み上げます。例えば**搭**の文字を選択すると「搭載の　トー」と読み上げるので、漢字や同音異義語を読み上げても、それが実際には何という文字なのかを把握することができます。

※実際の読み上げ方は、スクリーンリーダーによって異なることがあります。例えば文字を入力できる領域を選択した時に「エディット領域」と読み上げるものの他に「文字入力領域」や、「文字を入力して下さい」といった読み上げを行なうスクリーンリーダーがあります。

なお、付録 5『スクリーンリーダーで読み取った情報』に、本書序章の文章をスクリーンリーダーで読み上げさせた内容を収録しています。

また、スクリーンリーダーメーカーのホームページでは、スクリーンリーダーが読み上げる様子や開発理念といった情報が、より視覚的にわかりやすく紹介されています。

ただし、現状のスクリーンリーダーが読み上げられる情報は、**文字だけ**です。そのため、上述の画像に書いてある文字や、文字を画像として描画するウィンドウ、その他文字以外の情報は、スクリーンリーダーで読み取ることはできません。

またオペレーティングシステムで標準的に定義されていない機能を用いた画面では、スクリーンリーダーでアクセスすることができません。そのようなアプリケーションをスクリーンリーダーで利用できるようにするためには、独自の画面音声化の機能を搭載することが必要になります。この詳細は、第９章『音声読み上げを制御する』を参照して下さい。

**【 コラム 】 スクリーンリーダーの声いろいろ**

スクリーンリーダーが読み上げる音声の声質や高さ、抑揚の有無は、使用している音声合成エンジンによって変わります。従って、音声ナレーションで使われるような声を使えばきれいな発音になりますし、ロボットのような声を使えば、抑揚のない声で読み上げます。

また音声合成エンジンによって、日本語だけを読み上げられるもの、英語はフルスペルで読み上げるもの、英語を日本語英語で読み上げるもの、英語しか読めないものなどさまざまな個性があります。日本語に対応している音声合成エンジンの大半は、英語をフルスペルか日本語英語で読み上げます。

AUI ユーザー向けに開発されているアプリケーションの多くは、ユーザが声質や高さ、抑揚などを自在に選択できるようになっています。これは、どのような声で読み上げるのが良いかが、ユーザの好みや、音声を聞く環境によって変わるからです。

### ◆効果音

本書における効果音とは、ブザー音やコンピュータの起動音のような、スクリーンリーダーによる読み上げ音声以外の音のことです。

例えば、アプリケーションを使っている時に問題が発生すると、その内容がディスプレイに表示されます。それと同時に、スピーカーからブザー音が鳴るはずです。

聞こえてきた音は、「問題が発生しました」といった音声ではありませんが、ユーザは、「何か通常と異なる現象が起こった」ことに気づきます。さらにそこから「問題が発生した」という推定を行なうことができます。

AUI ユーザーにとっては、効果音が全く鳴らなくても、スクリーンリーダーが読み上げるのであれば、アプリケーションを使うことは可能です。
しかし AUI ユーザーは、パソコン操作のためにスクリーンリーダーの読み上げ音声を長時間、しかも繰り返し聞き続けなければなりません。このため、操作の結果を理解するのに時間がかかったり、見落としてしまうことがあります。
そこで有効なのが、**効果音を挿入する**ことです。音声の合間に効果音を挿入すると、ユーザは効果音という**音声と違う情報源**に気づくことができます。

『音声だけを用いてパソコンを操作している音声』および、『音声と効果音を用いてパソコンを操作している音声』を録音しました。
この中では、デスクトップ上の項目を探索した後、「スタート」ボタンを押し、「全てのプログラム」のメニューから「アクセサリ」を選択し、『メモ帳』を起動。そして「あえいうえおあお」という文字を入力した後メモ帳を終了するまでの流れを実演したものです。
音声だけを頼りに操作しているものでは、デスクトップ上でのカーソルの移動やアプリケーションの一覧がポップアップされた事、「メモ帳」が選択されたことが曖昧になっています。しかし効果音を鳴らすことで、それぞれの作業と、その結果どうなったのかがわかりやすくなります。

**◆キーボード操作**

多くの GUI ユーザーは画面に表示されているアイコンやボタンを選択する時にはマウスだけを使います。しかしそれらの操作は、キーボードでも代用することができます。

キーボードの内、マウスでの操作に近い役割を持っているキーは、上下左右の矢印キーおよび、エンターキーです。上下左右の矢印キーは実際に押した方向にカーソルを移動させることができ、エンターキーを押すことでその位置のアイコンやボタンをクリックすることができます。
その他にも、特定のキーの組み合わせを入力することで、実行できる機能があります。例えば複数のウィンドウに分割された画面で、別のウィンドウへカーソルを移動させるためには **Tab キー**や、**Ctrl + Tab キー**などを押します。
また、現在市販されているスクリーンリーダーの多くは、特定の内容をユーザに通知するためのショートカットキーを備えています。例えば、現在起動しているアプリケーション名を知りたい時、ユーザは「Ctrl + Alt + T」といったキーを入力すると、それをスクリーンリーダーが読み上げます。

実際にキーボードだけで操作できる事および、そのキーの組み合わせは、付録2「キーボードショートカット」に掲載しています。

このような仕組みがあるので、AUIユーザーはマウスを使うことなく、パソコンを操作することができます。ですから、ソフトウェアを開発する時には、必ずキーボードだけでも操作ができることを確かめるようにして下さい。

## 第２章『AUI と GUI の違い』

第１章で、AUI ユーザーは GUI に対して、画面を見る代わりにスクリーンリーダーの音声や効果音を聞き、そしてキーボードだけで操作していることを紹介しました。

このことから、同じアプリケーションを GUI ユーザーと AUI ユーザーが利用すると、画面に対する印象や具体的な操作方法に違いが生じます。

この章では、AUI ユーザーがアプリケーションをどのように操作しているのか、GUI ユーザーと比べるとどのような違いが生じるのか、その一部を紹介します。

---

### ◆AUI は GUI よりも時間がかかる

GUI ユーザー向けのアプリケーションは、ユーザに対して画像やアイコンのような視覚的な情報を提示します。これに対して AUI ユーザーは、その画像やアイコンに付属しているテキスト情報をスクリーンリーダーを用いて読み取ります。このため、AUI ユーザーが最終的に受け取るのは**スクリーンリーダーの読み上げ**という**音声情報**となります。

この視覚情報と音声情報は、**情報の形態**に大きな違いが有ります。視覚情報は、物の配置や色といった**空間的な情報**であるのに対して、音声情報は、ある時間の中で発せられた波形という**時間的な情報**です。

このことから、AUI というユーザインタフェースは、同様の操作を GUI を用いて行なうよりも時間がかかります。

しかも GUI ユーザーは画面全体を見渡すことで、同時に複数の情報を得ることができますが、AUI ではそれはできません。画面中のアイコンが何であるのかを同時に読み上げても、AUI ユーザーが全てを理解することができないからです。

AUI ユーザー向けのアプリケーションを設計する時には、時間に制限をかけることは適切ではありません。表示されているメッセージなどをユーザが読み終わる前に時間が過ぎてしまうからです。

もし、ユーザが制限時間内に操作を完了できるようにしたいのならば、制限時間を延長できるか、音声で読み上げる文字数を少なくすることが必要です。この具体的な方法や考え方については、第８章『音声化される情報をコントロールする』に記述しています。

### ◆AUI では聞きなおしが大切

前項で、AUI は音を聞くための時間がかかり、しかもユーザが一度に大量の情報を受け取ることができない点を述べましたが、このことによって、もう

一つ問題が生じます。それは、指示や情報を覚えていることが難しいことです。

人間は、見聞きした情報を記憶することができますが、**長時間**覚えておくことは困難です。しかし、AUI を使った場合、音を聞くために時間が必要なので、その分 GUI ユーザーよりも長い時間、ユーザは情報を覚えていなければなりません。

また GUI ユーザーは、仮に現在の画面表示を忘れてしまっても、ディスプレイに映っていればすぐに見直すことができます。これに対して AUI では、一度聞いた音を再度聞きなおすのにも時間がかかってしまいます。

このことから、過去に確認した情報は何度でも再確認できることが必要です。また同時に AUI ユーザー向けの情報は必要最低限のものとし、ユーザが覚える負担を少なくすることも有効です。

### ◆AUI ユーザーにとっての文字は表音文字である

前章で、スクリーンリーダーが文字を読み上げる時には、その音(おん)や訓、用例などを読み上げることを述べました。しかし、これで GUI ユーザーと同じように漢字や記号が使いこなせるわけではありません。

その大きな理由は、AUI ユーザーは文字を、**表音文字**として学習・使用しているからです。だから、AUI ユーザーは実際の漢字や記号の形を知っているわけではありませんし、同音異義語などの同じ発音の文字を間違えることがあります。

また同じ文字や単語でも、どのように読み上げるかは、スクリーンリーダーや前後の単語によって異なります。例えば **AUI** を、「エーユーアイ」と読み上げるもの、「アウイ」や「オーイ」と読み上げるものがあります。

このことから、文字の発音に注意する必要のある場合や、専門用語を解説する時には、ルビを付けるなどの対策を実施することを推奨します。この詳細は、第７章『ドキュメント』にて紹介しています。

### 【 コラム 】 「視覚障害」と「視覚障がい」

近年、**視覚障害**のことを「視覚障がい」と表記する動きがあります。これは、「障害者」という表記は社会的に適切でないという協議の影響を受け、自主的に変更されつつあるもので、実際には「視覚障害」と表記しても間違いではありません。

本書では、「視覚障害」と表記するようにしています。これは、スクリーンリーダー製品などが**障がい**を「さわりがい」と読み上げてしまうことによる誤解を防ぐためです。
なお多くのスクリーンリーダーは、ある単語に対してどのように読み上げるべきかを定義した辞書が設けられており、ユーザはそれを編集することで新しい単語を正確に読み上げさせることができます。

### ◆音声も長時間聞き続けると疲れる

GUI ユーザーが画面を長時間見続けると疲れるように、AUI ユーザーも音声を長時間聞き続けると疲れてしまいます。特に AUI では、**単調な音声読み上げ**によって疲れが出てしまいます。

AUI ユーザーは、スクリーンリーダーが読み上げる内容を頼りにアプリケーションを操作します。このため、ユーザのパソコンからは、同じ声質の音声が流れ続けます。
またスクリーンリーダーは、ユーザが何か一つ操作を行なう度に、その結果をフィードバックします。このため、同じ内容を何度も読み上げることがあります。
例えば本書には、「スクリーンリーダー」や「読み上げ」といった言葉が何度も登場します。この単語は著者がワードプロセッサで入力しているので、入力中の文字も含めて、頻繁にその音声を聞かなければなりません。

この結果、音声読み上げは単調になり、聞き疲れてしまいます。これを防ぐためには、効果音を用いたり、音のピッチなどに変化を付けることが有効です。
AUI は、ユーザが長時間アプリケーションを利用する可能性が高いので、このような工夫を持たせることはとても有効です。具体的な方法や注意点については、第６章『音』および、第９章『音声読み上げを制御する』に記述しています。

### ◆AUI は順次アクセス方式

GUI では、同時に複数のウィンドウやボタン、項目を表示することで、目的の機能が探しやすくなっています。画面全体の情報が一度に見える GUI の利点でもあります。
しかし、AUI ユーザーにはこれを代用できる機能はありません。全ての内容を同時に読み上げても困惑してしまうからです。
そこで AUI ユーザーは、現在アクセスしているウィンドウの項目からだけ情報を受け取り、その後次の項目、ウィンドウへアクセスするサイクルを繰

り返します。言い変えると、全てのウィンドウや項目の内容が記録されたカセットテープを早送り・巻き戻ししながら再生し、アクセスしている状態です。

この時に問題となることは、ウィンドウやボタンなど、文字ではない部分にアクセスした時です。スクリーンリーダーは文字しか読み取れないため、AUI ユーザーはこれらのボタンやウィンドウが何であるのか、わからなくなってしまいます。
このことから、全てのウィンドウやボタン、項目には、何を意味するものであるのかを記述することが重要です。例えば、クリックすることで操作を取り消すアイコンがあるのなら、そのアイコンには「キャンセル」という文字情報を持たせます。
スクリーンリーダーで読み取れる文字情報をウィンドウやボタンなどに設定する時に注意するべき観点は、第３章『ラベリング』に記述しています。

**◆AUI ユーザーはカーソルを移動させて画面を探索する**

前項で、AUI ユーザーは、ウィンドウやボタンを順次切り替えながらアクセスしていくことを述べました。この作業にはもう一つ重大な役割があります。それは、画面に**何が表示されているのかを探索する**ことです。

GUI であれば、画面を見れば、ウィンドウやボタンがどこにあるのか、といった状況を把握することができます。しかし AUI ユーザーは、画面に何があるのか、さらにその中でもどこにカーソルがあるのかを知ることはできません。
そこで、カーソルを移動させてスクリーンリーダーにウィンドウやボタンなどを読み上げさせます。これを繰り返すことで、画面がどうなっているのか、カーソルがどの辺りにあるのかを推測することができます。

しかし、ウィンドウやボタンが多過ぎると、それも探索しなければならないので時間がかかってしまいます。また覚える情報が増えるため、画面の構造を把握することも難しくなります。
この対策は、一度に表示されるウィンドウやボタンを少なくすることや、不必要な操作回数を減らせる仕組みを設けることです。この詳細は、第４章『画面デザイン』および、第５章『操作性』に記述しています。

また、ユーザの予想に反するカーソルの移動を行わないことも必要です。例えば右矢印キーを一回押した時には、カーソルは真右に一つ移動し、斜めに移動したり二つ以上先の項目が選ばれたりしてはいけません。

**◆AUI ユーザーの両手はキーボード上にある**

前章で紹介した通り、AUI ユーザーはマウスを使いません。このため、ユーザの両手は常にキーボードの上にあります。

またスクリーンリーダーは、いかなる状況でも読み上げ機能を提供しなければならないため、いくつかのキーを組み合わせることで機能を呼び出す**ショートカットキー**を多数搭載しています。

このようなことから、AUI ユーザー向けのアプリケーションでは、自由に両手を使えることを活かしたショートカットキーの組み合わせが多数あります。特に多いのは、「Ctrl」・「Shift」・「Alt」・「Windows」といったキーに、英数字のキーを組み合わせたものです。

また、AUI ユーザーの右手は、カーソル操作を行なう矢印キーや、クリックに該当するエンターキーの付近、左手はメニューバーにアクセスする Alt キーや別のウィンドウへアクセスする Tab キーの近くにあることがよくあります。これらの機能はパソコンを操作するために頻繁に利用しなければならないからです。

AUI ユーザー向けのアプリケーションを開発する時には、このユーザの手の位置を意識したショートカットキーを検討することをお薦めします。

ただし、全ての機能をショートカットキーだけに依存すると、多くのキー操作を覚えなければならず負担になります。また設定したショートカットキーと、スクリーンリーダーが搭載しているショートカットキーが競合し、どちらかの機能が使えなくなってしまうことがあります。

この対策として、メニューバーを用いても利用できるなどの、ショートカットキー以外の方法でのアクセスも可能にすることをお薦めします。この詳細は、第５章『操作性』に記述しています。

# 第２部　基本的 AUI ユーザビリティを学ぶ

この部では、AUI ユーザーが**全ての機能にアクセスできる**アプリケーションソフトウェアを開発するための原則と配慮について紹介します。

第３章では、全てのウィンドウやボタン、アイコンからユーザが正確に情報を受け取れるための方法および、その際の留意点について記述しています。
第４章では、AUI ユーザーがアクセスしやすい GUI をデザインするための方法および、その際の留意点について記述しています。
第５章では、アプリケーションを操作する過程でユーザが迷ってしまったり、操作の工程に行き詰まってしまわないために留意する点を記述しています。
第６章では、効果音を用いて情報を提示するアプリケーションが、AUI ユーザーにも有効に機能するための要点を記述しています。
第７章では、アプリケーションのマニュアルや、出力されたドキュメントデータが AUI ユーザーにも理解できるようにするための要点を記述しています。

## 第３章『ラベリング』

アプリケーションソフトウェアには普通、何かの利用目的のあるウィンドウやボタンが設置されています。これらウィンドウやボタンの利用目的を文字として明示することを、本書では「ラベリング」と呼んでいます。

AUI ユーザーにとっては、このラベリングには重要な役割があります。製品名、個々のウィンドウのタイトル、項目名といったラベルは、スクリーンリーダーが必ず読み上げる情報だからです。

例えば電子メールソフトには「受信メール」、「送信メール」、「メールアドレス」、「件名」といったようなラベルがあるのが普通です。このラベルがあることで、ユーザは今、どのウィンドウを見て、何をしようとしているのかを意識しながら作業を行なうことができます。

この章では、ユーザに意味の有る情報を提示するラベリングを行なうための要点について記述しています。

---

### ◆ウィンドウやボタンにはラベルを付けて下さい

アプリケーションソフトウェアで表示される全てのウィンドウ・ボタンには、そのウィンドウやボタンが何を意味するものであるのかを示すため、ラベルを付けて下さい。ラベルは、ディスプレイには表示されないこともありますが、スクリーンリーダーは必ず読み取って、AUI ユーザーにヒントとして提示するからです。

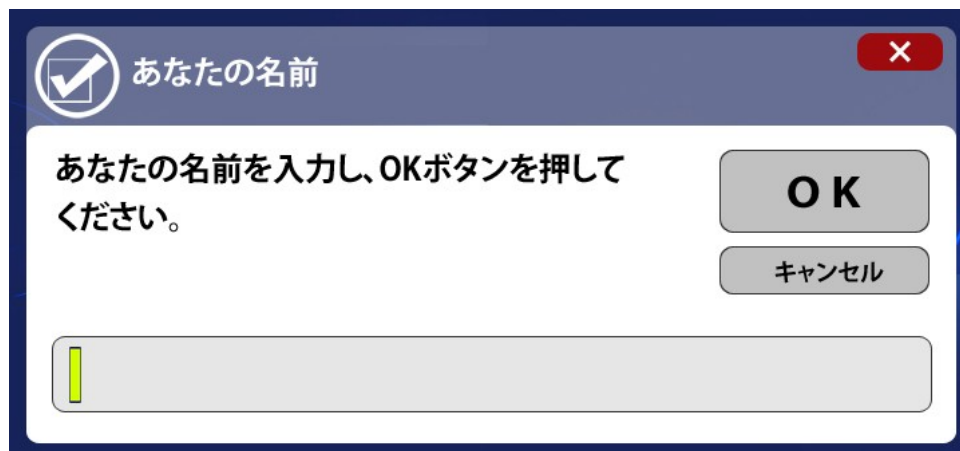
ラベルが設定されていないウィンドウやボタンにスクリーンリーダーがアクセスした場合、そのウィンドウやボタンが何をするものであるのか、AUI ユーザーが知ることはできません。

下記の画像は、エディットボックスに「あなたの名前」を書きこむことを促すウィンドウの事例です。

最初の画像は、ラベルを設定していないウィンドウです。スクリーンリーダーは、エディット領域があることしか通知しないので、AUI ユーザーは、名前の入力を求められていることを予測するのは困難です。

画像 3『ラベルの無いウィンドウ』

次の画像は、「あなたの名前」というラベルを設定したウィンドウです。スクリーンリーダーは、「あなたの名前　エディット領域」と読み上げるので、ユーザは、名前を入力する必要があることと、それを入力するための領域が選択されている、と予測することができます。

画像 4『ラベルのあるウィンドウ』

**【コラム】ラベルの無いウィンドウやボタンに対するスクリーンリーダーの代替手段**

現在、多くのスクリーンリーダーはラベルの無いウィンドウやボタンにアクセスした時、他のウィンドウや、前後に設置されている文字を AUI ユーザーにフィードバックしています。これは、実際にラベルの無いウィンドウに遭遇した時にユーザが困ってしまうのを防ぐためにスクリーンリーダーベンダーが行なった代替手段です。

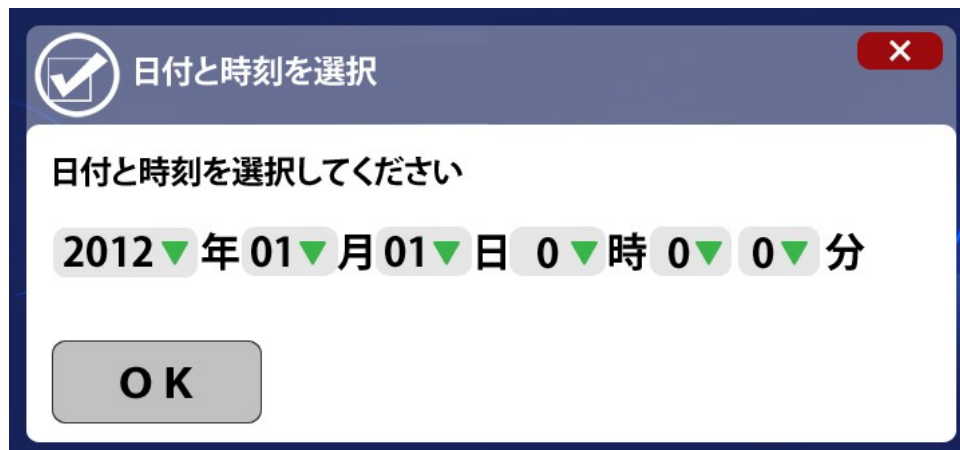
しかし、取得した情報が全て AUI ユーザーにとって有用であるとは限りません。

例えば日付と時刻の入力を促す画面では、日付や時刻となる数値を入力する領域の**直後**に「○○○○年　△△月　××日　○○時××分」などの日時の単位を表す文字が書かれていることがあります。

ここでもし、数値の入力領域にラベルが設定されておらず、スクリーンリーダーが、入力領域の**直前**の文字を代替情報としてユーザにフィードバックすると、個々の入力領域のラベル名が一つずつずれてしまいます。

具体的には、「月」を入力する領域のラベル名が「年」に、「時」を入力するラベル名が「日」になります。また「年」を入力する領域のラベル名は、直前に文字が無いので何もフィードバックしないか、もっと前に書かれている文字「登録時刻」といったラベル名になってしまいます。

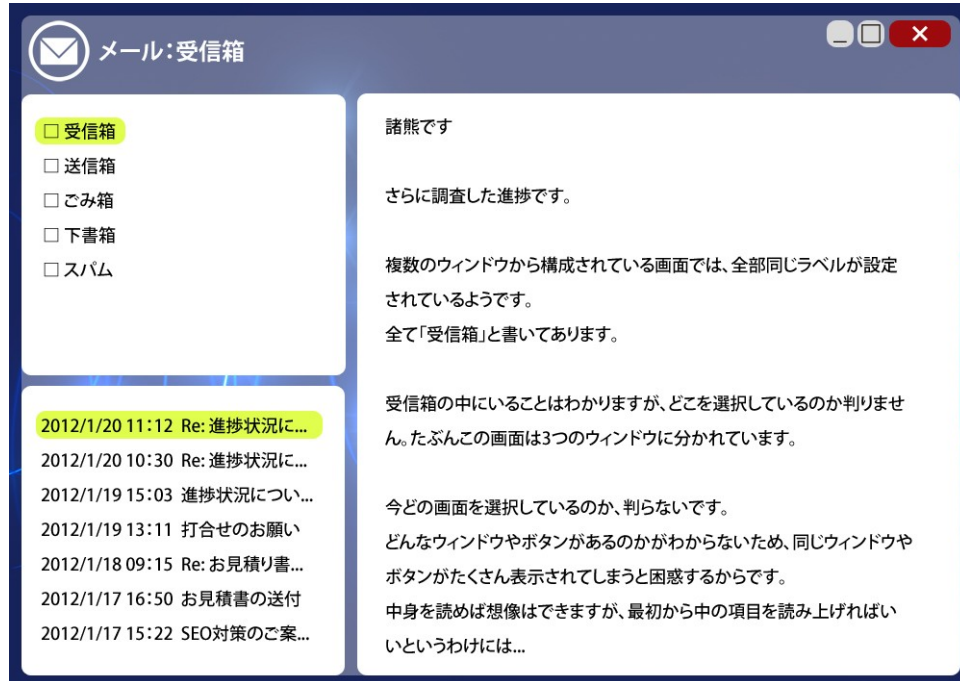
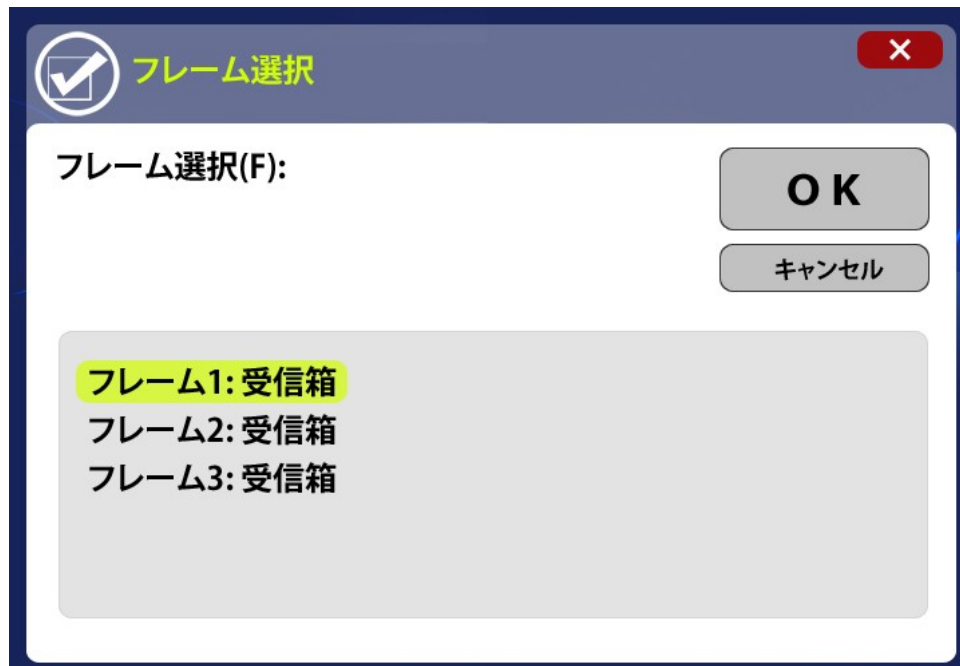
この対策は、該当する入力領域にテキスト情報を埋め込むか、「時」、「分」がどの入力欄に結びついているのかを定義することです。

画像 5『不適切な入力フォーム』

**◆異なるウィンドウやボタンに類似のラベル名は与えないようにして下さい**

GUI アプリケーションの多くは、複数のウィンドウで分割されており、個々のウィンドウ毎に行なうべき役割が区別されています。例えば以下の画面は電子メールソフトで、受信メールや送信メールを切り替えるウィンドウ、メールのリスト、メール本文などのウィンドウが表示されています。

このように、個々の役割が異なるウィンドウには、同じラベル名は付けないようにして下さい。AUI ユーザーは、実際にウィンドウを選択し操作してみるまで、どんなウィンドウやボタンがあるのかがわからないため、同じ名称のウィンドウ、ボタンがいくつもあると困惑してしまいます。

下記の画面はラベル名が全て「受信箱」に設定されています。このため、AUI ユーザーは、現在アクセスしているのは受信箱なのはわかっても、メール一覧なのか本文なのかがわかりません。

画像 6『同じラベルの 3 つのウィンドウ』

画像 7『ウィンドウに付けられたラベル』

**◆コンテンツ名など常に表示されるものは、ラベル名の末尾に付けて下さい**

ソフトウェア名やホームページの会社名などに当たる**コンテンツ名**は、常にウィンドウのタイトルバーに表示することが必要です。

「開くファイルの選択」や「保存」といった機能は、さまざまなアプリケーションで用いられているので、ただそれだけのラベルが書かれているのでは、AUI ユーザーは今どのアプリケーションのウィンドウを表示しているのかに迷ってしまうからです。特にユーザが何か他の作業を併用しており画面を切り替える操作を行なっている時には、このラベルが頻繁に読み上げられるため重要な情報源となります。

この結果、コンテンツ名と、現在実行している機能の 2 つのラベルを並べて表示することになります。この時の表示の順序について、現在アクセスしているウィンドウの表示が左側になるようにして下さい。

スクリーンリーダーはウィンドウ名を読み上げる時、必ず左から読み上げます。このため、ほとんど変化しないコンテンツ名を先頭に配置してしまうと、常にそれを読み終わるのを待たなければならなくなります。

例えば、「PDF リーダー」というアプリケーションで本書を表示している時、タイトルバーには「ソフトウェア AUI ユーザビリティガイドライン - PDF リーダー」のように表示するようにして下さい。

またこの PDF リーダーで本書を印刷しようとした時は、「印刷設定 - PDF リーダー」または、「印刷設定 - ソフトウェア AUI ユーザビリティガイドライン - PDF リーダー」のような表示を行なって下さい。

## 第４章『画面デザイン』

画面デザインは、GUI ユーザー向けのアプリケーションを開発する時には最も慎重に検討をする部分です。ユーザが画面のどの位置を見るのか、どんなアイコンやテキストがユーザにとって直感的であるのか、といった要素がアプリケーションの使いやすさに大きく影響するからです。

しかし、デザインを行なう時には、その画面を用いてユーザが何をするのか、何をしたいと考えるのかを意識しなければなりません。それを無視してデザインを行なうと、一見は優れていそうでも、実際に利用した時に不便さを感じてしまうからです。

例えば、ユーザがよく利用している機能を自動的にピックアップして、アイコンで表示する画面をデザインしたとします。この画面は、ユーザによって利用する機能の数が違うことや、見慣れてしまって見飽きることに配慮して、項目の数に合わせて各アイコンの配置が変わるようになっています。

このような画面は、利用可能な機能を全て表示した上でユーザが選択するリスト式の画面と比べると、一度に膨大なリストを見なくてもよくなるため個々の機能が選びやすくなります。特に利用可能な機能の内、実際に使う機能が少ない場合、余計な選択肢を見なくても良いため操作ミスを起こしにくくなります。

しかし、このような画面は AUI ユーザーにとってとても不便です。目的の機能アイコンが移動するため、どこへカーソルを移動させればいいのかがわからなくなるからです。

またこの画面は、全ての GUI ユーザーにとっても有用ではありません。ユーザは、よく使う画面のどこにどんなアイコンがあったのかを覚え、次に同じ操作を行なう時に参考にするので、アイコンの配置が頻繁に変わると戸惑ってしまうからです。

使いやすい画面をデザインする時には、まずその画面を通してユーザが行なおうとしている目的を明確にすることが必要です。そしてそのための最短の作業手順を作成し、それに従って画面をデザインしていきます。

この章では、特に AUI ユーザーがアクセスしやすく、またわかりやすい画面をデザインするための要点を記述しています。

**◆画面デザインは、システム標準のフォーマットを使用して下さい**

画面をデザインする時には、可能な限りオペレーティングシステムがサポートしているフォーマットを使って下さい。例えば Windows であれば、Windows で定義されているメッセージボックスやリストボックスが該当します。

過去、さまざまな製品が開発されてきましたが、それらの多くはこの標準化されたフォーマットを用いてデザインされています。それらと同じ外観や操作方法が応用できるのであれば、ユーザにあえて説明する手間を省き、またユーザも直感で操作方法を学習することが容易になります。

またスクリーンリーダーをはじめとする支援技術は、標準的な画面デザインが積極的に用いられているほど、アクセスが容易になります。標準化されている技術の情報は広く開示されているのでスクリーンリーダーにも反映することが容易であるからです。

ただし、画面に表示する文字が画像情報として処理されているような方法を用いると、スクリーンリーダーで一切読み上げることができません。そのような画面表示がシステム標準の機能になっていることもありますので、必ず確認して下さい。

もし、このような機能を使わなければならない時には、第 9 章『音声読み上げを制御する』を参照し、ユーザに音声情報を提供して下さい。

**◆画面の自動的な更新はユーザが制御できるようにして下さい**

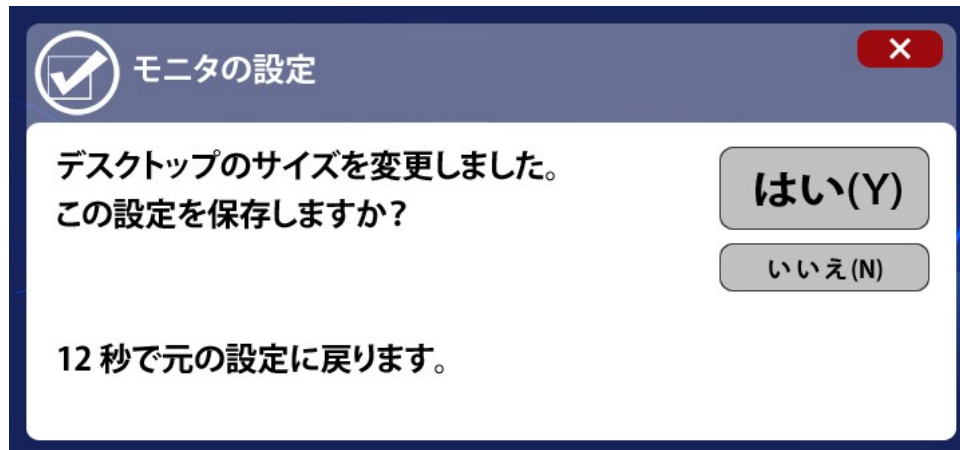
アプリケーションの中には、自動的に最新の情報を取得して表示する、動的な機能を持つものがあります。例えばインターネットチャットを行なう時には、一定の間隔で新しい発言がされていないかどうかを確認します。

しかし、自動的な更新を行なう機能は、ユーザがその更新ペースに追いつけることが必要です。まだ画面上の内容を読み終えない内に次の画面に移動してしまうと、ユーザは困惑してしまいます。

これは、特に AUI ユーザーにとって重大な問題になります。AUI は音声を聞くための時間が必要なので、音声で読み終わる前に情報が変わってしまう可能性があるからです。

下の画像は、パソコンの画面解像度を変更した時に表示される画面です。メッセージの通り、12 秒以内に「はい」ボタンを押さなければなりません。しかし、メッセージが長過ぎるため、AUI ではこの指示を理解し「はい」ボタンを押す前に 12 秒経ってしまいます。

AUI ユーザーがこれを操作する時には、やり直しすることを前提にし、一回目では表示されているメッセージを読み取ることだけを行ないます。その後読み終えたら改めてこの画面を表示し、設定を完了しなければなりません。

画像 9『12 秒以内に設定を変更するよう呼びかけるメッセージ』

このような問題を防ぐ方法は、自動的な更新が行われる間隔や、更新するかどうか自体を、事前にユーザが制御できることです。例えば更新するまでの時間を 15 秒、30 秒、60 秒、手動更新から選べる、といった工夫を設けることが有効です。

またこの設定は、実際に対象の操作を行なう前の段階で設定できることが必要です。自動的に更新されるまでの時間を、画面が更新されている間に切り替えようとしても、その手順を理解する時間が無いかもしれないからです。

ただし、インターネットチャットのようなリアルタイム性が求められるアプリケーションでは、更新までの感覚が極端に長くなると、リアルタイムでなくなります。この対策は、受信する情報から不要な内容を削ってユーザに提示することや、過去の画面情報を後から参照できるようにすることです。

ユーザに提示する情報に優先度を付ける際の規準や、繰り返し情報を参照できる利点については、第８章『音声化される情報をコントロールする』にて紹介しています。

**◆ウィンドウの内容を更新したら、音でも通知して下さい**

アプリケーションの中には、自動的に最新の情報を取得して表示する、動的な機能を持つものもあります。例えば電子メールソフトには、定期的に新着メールが届いていないかどうかをチェックし、もし見つかったら自動的に受信する機能が搭載されています。

しかし、ユーザが画面を見ていなかった時に情報が更新されていたり、更新された内容が少なかったりすると、ユーザはそれに気づかない可能性があります。

またスクリーンリーダーの多くは、ウィンドウやフォーカスの変化をユーザに通知することはできますが、ウィンドウ内のテキストの部分的な変更や、フォーカスされていないウィンドウ内の情報の変化を通知することはできません。従って、後者のような情報の更新を実施した場合、AUI ユーザーはその変化に気づくことはありません。

そこで、ウィンドウの内容が更新されたことや、何が変化したのか、といった情報を音声や効果音で通知することが有効です。

この仕組みは、AUI にとっては基本的であり、とても重要なことです。AUI ユーザーは、操作の結果を音声で受け取って初めて次の操作へ移るので、操作結果の音声は全て、**自動的な情報の更新**と似た意味を持っているとも言えるからです。

またこれは、複数の作業を同時に行なっているユーザにとっても有用です。例えば新着メールが届くのを待ちながらインターネットで情報収集をしている時、メールが届いたことが音で通知されるようにすれば、ユーザはインターネットに専念することができます。

ただし、音はユーザにとって邪魔になることもあります。特に AUI ユーザーは、スクリーンリーダーの音声が聞こえなくなるような音が鳴ると、AUI が使えなくなるので操作を続行できなくなります。

このことから、音を再生する時には同時に、音を止められる方法も提供することが必要です。

またその方法は、指定した位置のボタンをスクリーンリーダーからの情報を頼りに探せるだけでなく、現在の状況に関係なく「Escape」や「Ctrl+S」のようなキーを押せば即実現できるものを推奨します。これはスクリーンリーダーの音声が聞こえていないことが前提となるので、スクリーンリーダーでボタンを探す操作を強制しても見つけられないからです。

**◆一つ以上前の画面内容を参照できるようにして下さい**

アプリケーションを利用していると、直前の画面に戻りたくなる場合があります。

例えばユーザが離席し戻ってきた後、どこまで作業を進めていたのかを確認したくなることや、画面内容が自動的に更新された時、どの部分が変更されたのかを更新前の画面と比較したくなります。

このような時には、以前の画面内容を確認できる機能があると有効です。
またこの機能があることで、ユーザが誤操作で次の画面へ進めてしまった時、すぐにやり直しをすることができるようになります。

なお、前の画面を参照するための機能では、2 画面以上前の内容を参照してはいけません。これは、2 画面以上前の内容を参照したいと思った時には、一つ前の画面に戻る操作を 2 回行なうことの方がわかりやすいからです。

### ◆過剰なウィンドウやボタンを置かないようにして下さい

GUI デザインでは、ディスプレイ上にいくつもウィンドウやボタンを表示することができ、個々の画面の状況を、容易に確認することができます。

しかし、ウィンドウやボタンがあまりにも多いと、ユーザはどこから見れば良いのかに戸惑ってしまったり、操作ミスを起こしやすくなってしまいます。

また AUI ユーザーの場合、ウィンドウやボタンが合計何個あるのか、またそれぞれどんな内容であるのかを知るために、一つずつアクセスして確認しなければなりません。このため、ウィンドウやボタンが多いほど、その確認作業は難しくなります。

このことから、一画面内でのウィンドウやボタンの数は少なくなるようにして下さい。ウィンドウやボタンを少なくするためには、関連性はあるが頻繁に変更する必要のない項目を異なる設定画面に移動したり、類似した項目をグループ化して階層化することが効果的です。

例えばワードプロセッサで、文字に引く線の種類を選択する場合には、下線・上線・打ち消し線などを一つのグループにします。さらに線の色や波線・二重線などの形状を細かく設定するならば、「下線の設定」というボタンを設け、それをクリックすることで新しいウィンドウがポップアップすることが有効です。

### ◆直接操作する頻度の少ないアプリケーションは最小化できるようにして下さい

アプリケーションの中には、ユーザが操作している時間が多くないものがあります。例えばアンチウィルスソフトウェアはユーザがパソコンを起動してから終了するまでの間常に起動していますが、ユーザが常に何か操作しているわけではありません。

このようなユーザが直接操作する頻度が少ないアプリケーションのウィンドウは、必要に応じて非表示にできることが必要です。余計なウィンドウが画面を埋め尽くすと、本当に行ないたい作業のウィンドウが探しにくくなるからです。

特にAUIユーザーは、現在アクティブにしているウィンドウ一つにしかアクセスすることができないため、ウィンドウの数が多いほど、目的のウィンドウを切り替える操作は困難になります。

Windowsには、この機能として**最小化**ボタンが設けられています。最小化ボタンを押すと、押されたウィンドウは小さくなり、ツールバーへ移動します。このウィンドウは、再びクリックすれば復元できるようになっています。

画像8『ツールバー』

## 第５章『操作性』

操作性の優れている製品はとても魅力的です。なぜならば、多くのユーザは製品に対して、**使い続けていきたい**という希望を持って接しようとするからです。

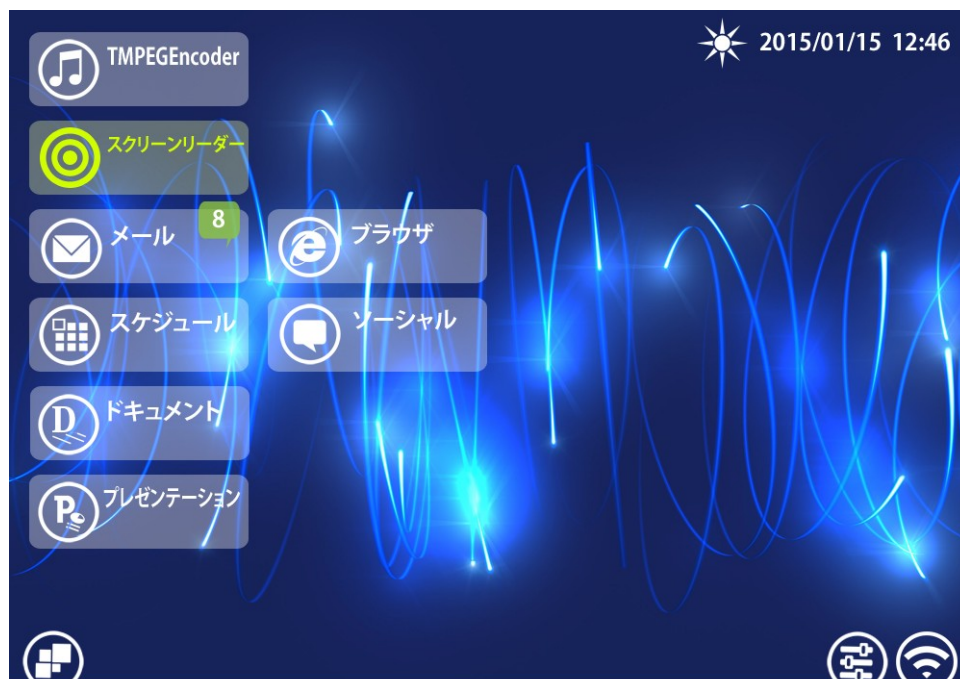
この章では、特に AUI ユーザーとキーボード操作に念頭を置いて、優れた操作性を提供するための要点を記述しています。

---

### ◆特定の操作を行なった時の挙動は常に一対一対応するようにして下さい

ユーザはアプリケーションに対して、ある操作をすると、一定の結果が得られることを予想しながら操作しています。この期待通りに動作することは、ユーザが感じる負担を減らし、スムーズにアプリケーションを使いこむのに有効です。

下の画像は第１章で紹介したデスクトップ画面です。この画面で上矢印キーを押すと、一つ上の項目である「TMPEGencoder」が選択されます。その後下矢印キーを押すと、元の位置「スクリーンリーダー」が選択されます。

画像１『デスクトップの画面』

このような、一定の操作を行なった結果が常に一定であることは、AUI ユーザーがアプリケーションを操作できるためにとても重要なことです。

AUI ユーザーは画面全体に何があるのか、今何を操作しているのかを知るために、画面のさまざまな場所へカーソルを移動させ、探索します。そして、

それぞれの画面位置と、そこへカーソルを移動させるための手続きを関連付けることで学習しています。

このため、ユーザが想定している動作と違うカーソル移動をすると、それまで関連付けられていた手続きが複雑になり、理解が困難になります。また操作を繰り返している内に、それまでどのように操作していたのかを見失う、迷子状態に陥る可能性があります。

例えば下の表のように、デスクトップ上にさまざまなアプリケーションのアイコン(表ではテキストのみ記した)をバラバラの位置に配置します。これをAUIユーザーが操作すると、カーソルが突然2つ分移動したり、斜めに移動したりします。

**表2『デスクトップにアイコンをバラバラ配置』**

| | | | |
|---|---|---|---|
| マイ ドキュメント | スライドショー | | 音でコマンドライン |
| マイ コンピュータ | セキュアアンチウィルス | ワープロ2011 | |
| ごみ箱 | | ドロップ圧縮 | 本日の日程 |
| CDプレイヤー | フォト・アルバム管理2012J | | 電卓 |
| | | 路線探索 | |
| E-Mail life | 指紋認証 | | |
| InternetReader | | DVD Write | |
| TMPEGEncoder | | 諸態作業フォルダ | |
| スクリーンリーダー | 共有用フォルダ | | |

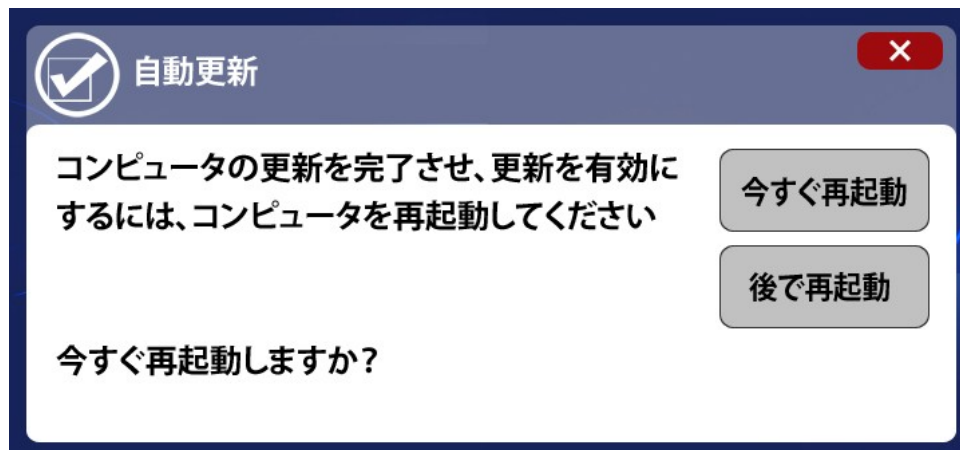
**◆他の作業をしているユーザを邪魔するような動作をしないで下さい**

いくつかのアプリケーションソフトウェアを同時に使用しているユーザの多くは、必要に応じてウィンドウを切り替えながら作業を進めています。しかしこの時、操作していないはずのアプリケーションが突然ポップアップしたり、キー操作に反応したりすると、戸惑ってしまいます。

ユーザがアクセスしていないアプリケーションは、指示を無視して動作しないようにして下さい。ユーザの**指示**とは、そのアプリケーションのウィンドウにアクセスすることや、該当のアイコンをクリックすることです。

下記の画面は、Windows のシステムソフトウェアの自動更新です。
ソフトウェアを更新したことで再起動を促すメッセージを表示するのですが、初期設定では、それがいつ表示されるのかを AUI ユーザが知ることはできなくなっています。
このようなウィンドウのポップアップは、キーボードを利用している時に特に大きな問題に繋がります。
例えば文字を入力し、確定しようとしてエンターキーを押そうとした直前にこの画面がポップアップすると、文字入力の確定させるつもりで押したエンターキーによってパソコンが強制的に再起動してしまいます。この時のユーザは文字入力のつもりで手を動かしているので、新しいウィンドウのポップアップによって音などが鳴ったとしても、指を静止させることは困難です。

画像 10『再起動を促すウィンドウのポップアップ』

自動更新
コンピュータの更新を完了させ、更新を有効にするには、コンピュータを再起動してください
今すぐ再起動しますか?
今すぐ再起動
後で再起動

もし、このようなポップアップが必要な時には、その初期値としてキャンセル操作や、保留とする操作が選ばれるようにして下さい。これによって、ユーザが操作を誤ってしまった時にも、その被害を受けることを防ぐことができます。

### ◆目的達成のために複数の操作手段を提供して下さい

ソフトウェアは、特定のデバイスだけで操作されることを前提としてはいけません。マウスだけ、またはキーボードだけで操作できるということは、そのどちらかが故障した時や、ユーザが利用できない状況に陥った時、使用不能になってしまうからです。
例えば USB マウスやワイヤレスマウスを使っている GUI ユーザーは、USB ポートの接触不良や電池切れなどが原因で操作を受け付けなくなった時、戸惑ってしまうかもしれません。

その一方、AUI ユーザーの多くはマウスを使えません。このためアプリケーションがキーボードで操作できなければ、そのソフトウェアを使うことは困難になるか、全くできなくなります。

ただし、キーボードで操作できたとしても、ショートカットキーが押せなければ使用できない機能だけでアプリケーションを構成することは避けて下さい。設定したショートカットキーが、ユーザが常用している異なるアプリケーションのショートカットキーと競合することがあるからです。

特に AUI ユーザーの場合、アプリケーションに割り当てたショートカットキーがスクリーンリーダー固有のショートカットキーと競合することがあります。この場合どちらかの機能が利用できなくなるため、例えばユーザはスクリーンリーダーの動作を一次的に停止させるなどの対応をしなければならなくなります。

このような問題を解決するための方法の一つとして、プルダウンメニューやリスト式メニューも併用することを提案します。このようなメニューは、利用可能な機能を一覧し、選択する操作が、マウス、キーボードのいずれでも利用でき、また階層構造を表現できるので、情報を整理するのにも適しているからです。

**【 コラム 】 スクリーンリーダーによるマウス代行操作機能**

近年のスクリーンリーダーは、特定のショートカットキーを入力することでマウスカーソルを移動させたり、クリックさせることができるものがあります。またマウスカーソルで選択した位置にある情報を音声で読み上げられるものもあります。

しかしこの機能があるから、AUI ユーザーがマウス操作をできるわけではありません。マウスカーソルを移動させたりクリックできる手段があっても、現在どの位置にカーソルがあって、移動させるべき場所がどこなのかがわからないからです。

また自動的にマウスカーソルをボタンの上などに移動させられる機能があるとしても、同じことが一般的なキーボード操作で実現できれば、その方が AUI ユーザーに喜ばれます。これは、一連のマウス代行操作機能は特別なショートカットキーによって実装されているため、一般的なキーボード操作に比べてその手順が難解だからです。

一般的なキーボード操作については、付録 2「キーボードショートカット」を参照して下さい。

**◆ショートカットキーの割り当ては、アプリケーション共通の機能を優先して下さい**

ショートカットキーは、ソフトウェアの特定の機能に素早くアクセスするために用いられています。この中でも、コピー(Ctrl+C)・ペースト(Ctrl+V)・ファイルの保存(Ctrl+S)など、アプリケーション間で共通のキーが割り当てられている機能があります。

特定の機能にショートカットキーを割り当てる時には、このような一般的に使われているショートカットキーと関連付けられている機能があれば、同じキーを割り当てるようにして下さい。これによって、ユーザが特定の目的を達成するために、直感した機能を素早く実行できるようになるからです。

さらに、そのようなショートカットキーと対応する機能については、説明書に詳細を書くことも必要ではなくなります。ユーザは、機能とショートカットキーを見るだけで、何ができるのか、どのキーを押せばいいのかをすぐに理解できるからです。

アプリケーションで共通に利用されているショートカットキーと、それに関連付けられている機能の一覧は、付録2『キーボードショートカット』を参照して下さい。

逆に、アプリケーションで共通に利用されているショートカットキーに、全く違う機能を割り振ることは避けて下さい。例えば「Ctrl」+「X」キーに「アプリケーションを終了する」という機能を割り当てて、文字列のコピーやカット機能が存在していると、ユーザは誤ってこの操作をしてしまうかもしれません。

また「Ctrl」+「V」と「Ctrl」+「Shift」+「V」のように、よく似たキーを用いる時には、それに近い機能を割り当てるようにして下さい。例えばこの場合は、「文字を貼り付ける」と、「異なる書式に変更した文字を貼り付ける」のように、本来の機能を修飾するようにします。

**◆同一の操作手順を繰り返す必要がある時には、その繰り返し回数を減らす仕組みを設けて下さい**

携帯電話でホームページを閲覧すると、画面をスクロールするのに同じ操作を頻繁に繰り返すことに苛立った挙句、行き過ぎてしまい戻るはめになった、といった操作ミスを起こした経験があるはずです。

このような繰り返し行なう必要のある操作がある場合には、その繰り返し回数を減らすための仕組みを持たせることが効果的です。例えば、一度の操作で、10回分の操作ができれば、操作回数を大幅に減らすことができます。

Windowsでは、画面スクロールの効率を高めるためのショートカットキーとして、「PageUp」、「PageDown」、「Home」、「End」の4つのキーがあ

ります。「PageUp」と「PageDown」キーは画面 1 つ分または上下矢印キー10 回分、「Home」と「End」キーは、先頭と末尾への移動を瞬時に行なうことのできるキーです。

ただし、PageUp キーや PageDown キーで複数回分カーソルを移動させる時には、その移動量は常に一定になるようにして下さい。カーソルを移動する度に、その移動量が 10 個、6 個、8 個などと変化していると、「一定回数繰り返せば目的の操作が完了できる」というユーザの期待に合致しなくなるからです。

**【 コラム 】 WEB ページの「本文へ移動」リンクの意義**

近年、WEB ページの中で「本文へ移動」のようなリンクが先頭に設置されているサイトが増えてきました。これは、「ブロックスキップ」と呼ばれる WEB アクセシビリティに配慮したページ制作手法の一つです。

WEB アクセシビリティの JIS 規格*『JIS X8341-3 達成規準 7.2.4.1』*には、クリックすると、本文セクションへカーソルを移動できる仕組み**ブロックスキップ**機能を WEB ページに持たせることを求める記述があります。

このブロックスキップが無い場合、AUI ユーザーは常に WEB ページを上から順に読み進めなければなりません。このため、本文を把握するのに時間がかかるだけでなく、そもそもどこからが本文なのかわかりにくくなってしまう、という問題が生じます。

同様にブロックスキップは、携帯端末のようなディスプレイの小さな環境で閲覧しているユーザにも効果的です。ディスプレイを繰り返しスクロールし続ける必要がなくなるからです。

ただし、このブロックスキップでは注意しなければならないことがあります。それは、画面のスクロールと違って、スキップした文書中にある内容全てをユーザは読めないことです。

このことから、スキップの対象となる内容が更新され、しかも記載されている内容がユーザにとって重要である場合、その WEB ページでは、ブロックスキップを置くことは適切ではありません。

例えばショッピングサイトのトップページのように、ヘッダ・本文セクションの区別が曖昧で、しかもコンテンツが頻繁に変わる可能性がある場合、ブロックスキップを置く意味はほとんどありません。一方で、商品検索画面のように、「検索結果」というユーザが見たいコンテンツが明確なページでは、ブロックスキップを設けることは有効です。

**◆ショートカットキーは、ユーザが手を押しやすいように割り当てて下さい**

特定の機能にショートカットキーを割り当てる時には、実際にそれを操作しているユーザがどのように手を動かしているのかに着目して下さい。そして、押しやすくなっているかどうかに注意して下さい。

例えばマウス操作が主体で右聞きのユーザが使うアプリケーションを制作する時には、ショートカットキーは左手で押せると使いやすくなります。これは右手が常にマウスを握っているので、両手で押すショートカットキーがあると、その都度マウスを手から離さなければならなくなるからです。

これに対して AUI ユーザー向けのショートカットキーでは、マウスを必要としません。しかし、同じキーを繰り返し押す操作が多いため、頻繁に手が移動するショートカットキーを割り当てることは避けるべきです。

AUI ユーザー向けのショートカットキー割り当ての際には、右手は矢印・エンターキー付近、左手は S・A キーの付近に集まるように考えてみると効果的です。これは、AUI ユーザーがコンピュータを操作するにあたって、頻繁に押すキーであることによるものです。

AUI ユーザーはまず**ホームポジション**を学習します。ホームポジションとは、キーボードの「A」・「S」・「D」・「F」・「G」・「H」・「J」・「K」・「L」・「Semicolon」のキーが並べられている位置です。

ホームポジションを中心に指を移動させることで、各々のキーにアクセスするための時間を均一にすることができます。特に AUI ユーザーはキーボードを中心に使うので、この位置を学習することが有効です。

またホームポジションには、人差し指で触れられる位置「F」・「J」のキーに突起が付けられているので、どんなキーボードでも、触覚を用いてホームポジションを探し当てることが容易です。

次に AUI ユーザーは、矢印キー、エンターキー、左 Alt キー、左 Windows キー、そしてタブキーを覚えます。

矢印キーは、マウスを使ってカーソルを移動させること、そしてエンターキーは、マウスでクリックすることと同じ働きをするので、Windows を操作するためには、この二つのキーは欠かすことができません。

一方、左 Alt キーは、メニューバーにアクセスするために、そして左 Windows キーは Windows のメニューにアクセスするために使います。また Windows では、複数のウィンドウやダイアログボックスの間を移動するためにタブキーが利用されます。

このような理由で、AUI ユーザーの手はホームポジションと左 Windows やタブキー付近、矢印やエンターキーの付近とを頻繁に行き来しています。

しかし実際には、ホームポジションは文字を入力するためにしか使わないので、操作に習熟したユーザほど、手の位置は徐々にホームポジションから外側に離れていきます。

この結果、AUI ユーザーにとって押しやすいキーは、Ctrl キーや矢印、ホームポジションの手前側のキーとなります。そして、キーが奥に行くほど、中央に寄るほど押しにくくなる傾向があります。特に F5～F8 キーは最もユーザの手から遠い位置にあるため押しにくくなります。

ただし、右 Windows キーや右 alt キーを、ショートカットキーとして使うことを前提にしてはいけません。理由は以下のように 3 つあり、ユーザが困惑してしまうからです。

1. Windows キーや Alt キーを使う操作の多くは、何らかのメニューを表示する時なので、右手は項目選択を行なう矢印キーの位置に移動します。このため右 Windows キーや右 Alt キーを指定すると、ユーザの右手は余計に行き来しなければなりません。
2. 現在のパソコン指導の現場では、右 Windows・右 Alt キーはパソコンによって配置が違う問題を考慮し、優先的に左 Alt キー、左 Windows キーから指導するようになっています。このため、ユーザは Windows キーや Alt キーの名前を聞くと、左側のキーを最初に連想します。
3. パソコンやキーボードの中には、右 Windows キーや右 Alt キーが存在しないハードウェアがあります。このため、それらのキーを押すショートカットキーに遭遇すると、左手で押さなければなりません。

## 第６章『音』

本書における**音**とは、クラシックのような音楽ではなく、ユーザに何かの情報を提供するために用意された音声や効果音の情報です。例えば、アプリケーションソフトウェアの中で、作業が完了したことを知らせるために**ベルの音**を使った場合、この音は常に何かの作業が完了した時、すぐに鳴らすことが必要です。

音は、画像や文字の情報とは全く異なる長所と短所を持っている媒体です。それらの特長を理解して音をアプリケーションに組み込むことで、AUI ユーザビリティはより高くなります。

この章では、音の情報を提示するアプリケーションをデザインする時に心に留めておいて欲しい事柄をまとめています。

---

### ◆音の長所・短所を理解して下さい

アプリケーションに音を組み込む時には、まず音の長所と短所について考えることが必要です。音は、ソフトウェアのデザイン・操作性の両方に影響を与える媒体なので、極端に悪い使い方をするとユーザに敬遠されてしまいます。

音の情報は、画像情報と比べて、以下のような長所があります。

1. 状況の経過や終了の確認を音で表現すれば、それを確認するためにウィンドウへアクセスする必要がなくなります。
2. 同じ画面を見続けることで、疲労したり画面の変化を見落とす問題を防ぐことができます。
3. 効果音などが一切無い**無音**状態を実現できるので、画像情報よりもわかりやすいフィードバックができます。

一方、音には以下のような短所があります。

1. 音は、鳴らす場面やその音量の組み合わせが適切でない時に、ユーザにうるさい、または全く聞こえないという印象を与えます。
2. 複数の音が混在している時、個々の音の違いを区別することが困難になります。
3. 音声情報は、それを聞く時間を必要とするので、大量の情報を短時間で提供することはできません。

### ◆音を扱うアプリケーションでは、音量調節機能を持たせて下さい

音楽プレイヤーや、スクリーンリーダーのような、音の情報が操作の基盤となるアプリケーションでは、音声や効果音などの音量はユーザがいつでも変更できることが必要です。

AUI ユーザーは、スクリーンリーダーを使ってアプリケーションソフトウェアを操作します。この時、アプリケーションソフトウェアからも音が鳴っていたら、相互の音量バランスを調節できなければ、どちらかの音が聞こえなくなってしまいます。

この時、音量変更によって、Windows のマスター音量などのパソコン全体の音量が変更されることは避けて下さい。この**変更**には、一時的に音量を変更し、アプリケーションの終了時に元の値に変更することも含まれます。

例えばスクリーンリーダーを使いながら音楽を聞いていて、「音楽が大きいことでスクリーンリーダーの音声が聞きにくい」とします。そこでスクリーンリーダーの音量を上げるのですが、この時パソコン全体の音量を変更してしまうと、同時に音楽の音量も上がります。これでは、「音声が聞きにくい」という問題を解決できません。

**◆ユーザに情報の変化を通知するための音は、再生までのタイムラグを最小限にして下さい**

音声や効果音に該当する音のデータの多くは、Wave や MP3 といった形式のデータとして保存されており、再生するためにはメモリ上にロードする必要があります。また音声読み上げを行なう時には、合成音声をメモリ上にロードして再生することで、初めてユーザが聞くことができます。

そのため、実際に音を再生するまでには時間がかかります。どれほど時間がかかるのかは、使用する音声データや、読み上げる文字の長さなどにより大きく変動します。

このような音を、ユーザに情報を提示する目的で使う時には、再生までのタイムラグは少なく、そして常に一定であることが重要です。例えば作業を完了したことを知らせるベルの音は、作業が完了したらすぐに鳴ることが必要ですが、それの再生が遅れると、ユーザは**故障したのか？**と戸惑ってしまいます。

これを実現するための工夫は、例えばアプリケーション起動時に、再生する効果音を予めメモリ上にロードしておくことや、デコードの必要な MP3 を使わない、といった方法があります。

同様に音声ガイダンスを組み込む際にも、その音声を頼りに操作するものであるなら、**ガイダンスまでに時間のかかる高音質の音声**よりも、**低音質でもタイムラグの少ない音声**を使うべきです。高音質の音声は大量のデータを制御するため、1 分間で 50 回も音声合成を行なう AUI で使うと、ソフトウェアの安定した動作にも支障が生じます。

また音声合成は、1000 文字を超えるような大量の文を一度に読み上げさせると、同様に時間がかかってしまいます。このような時には、一段落、一文ずつ区切って読み上げさせるようにすると有効です。

**◆効果音として使う音には変化を付けて下さい**

音は、ユーザが何かの作業を行なう度に、その結果を素早くフィードバックすることに有効です。この時、効果音の音量や定位、種類を変化させると、複数の異なる情報を提示する場合に、個々の違いがより区別しやすくなります。

例えば、作業の進行度合いを音で示す時には、その進行度合いによって音の高さが高くなっていくようにすると、進行度合いの数字を見なくても、現在の状況を予測することができるようになります。

『進行度合いを音と読み上げで伝える様子』これは、作業中定期的にビープ音が鳴るのですが、進捗が進む毎にこのビープ音が高くなっていきます。また定期的に、進捗状況をパーセンテージで読み上げます。

変化する情報が、ある特定の数値である時には、音の高さを変えることが有効です。例えば、いくつかのデータの中で最も秀でた数値や、作業の進行度合いを数値化した時、音の高さを変えることは適しています。

表や画像の特定位置のデータを参照した時の現在位置の情報を通知する音は、音の定位や奥行感を調節することが有効です。例えば左上にある情報であれば全体的に音量を小さく、さらに左音量は右音量より高く設定します。逆に右下にある情報を参照する時には、右側からだけ音が鳴るようにします。

もし、単に伝える情報の種類が違うことをユーザに通知する場合には、異なる効果音を再生して下さい。例えば、システムがユーザの操作を受け付けない時の音では、その操作がアプリケーションで対応していなかった場合と、操作には対応しているが、これ以上受け付けることのできない時とは、異なる音でなければなりません。

なお情報の変化には、カーソルの形状の変化や発声する入力イベントの違いも含まれている点に注意して下さい。例えばキーボードやマウスを操作しカーソルをアイコンに重ねた時に音を鳴らしたのならば、逆にアイコンからカーソルが離れた時にも異なる音を鳴らすことが必要です。

また音の定位を変える時には、その音はステレオヘッドホンなどを前提としてしまう点に注意して下さい。また同じ音では個々の違いが区別しにくいという短所があるため、音の定位を工夫しても、必ずしも正確に操作できるとは限りません。

**◆アプリケーションは、長時間、繰り返し音声を聞き続けることを前提に開発して下さい**

AUI ユーザーはスクリーンリーダーを用いてパソコンを操作しますが、もしパソコンを 2 時間操作し続けたとしたら、ユーザはその間、スクリーンリーダーの音声を聞き続けたことになります。

また多くのアプリケーションは、いくつかの操作画面を経て作業を遂行しますが、最後には最初に表示した画面に戻ってきます。つまり、同じ音声情報を聞くことが頻繁にあります。

アプリケーションを制作する時には、このような長時間音声を聞き続けることや、同じ音を何度も聞くことを想定するように心がけて下さい。それも、ただ聞き流すのではなく、アプリケーションを操作するためである点に注意して下さい。

一般的にユーザは、同じ音声を聞き続けると、その音声に慣れてしまうので、少し異なる情報と混同してしまうことがあります。また長時間聞き続けると疲労し、徐々に音声を聞くことへの集中力が低下していきます。

これは、常に AUI に頼っている視覚障害ユーザーにも例外ではないため、とても重大な問題となります。

この対策として有効なのは、音声読み上げなどを途中で止められるようにすることや、効果音を織り交ぜて、音の情報に変化を付けることです。また長い時間音声を聞き続けなくてもいいように、不必要な情報を読み上げさせない、といった工夫も大切です。

## 第７章『ドキュメント』

本書で言うドキュメントとは、アプリケーションの使い方を記した説明書や、ウィンドウに表示されるテキスト、出力されるファイルなどを総称したものです。AUI では、スクリーンリーダーなどがこれを読み取るため、極めて重要な情報源となります。

この章では、AUI ユーザーにわかりやすいドキュメントを提供するための配慮について記述しています。

---

### ◆プレーンなテキストだけでも情報が伝わるようにして下さい

説明書などのドキュメントには、テキストや画像、音楽などのさまざまなマルチメディアコンテンツを含めることができます。しかし、提示したテキスト以外のマルチメディアコンテンツを、全てのユーザが理解できるとは限りません。

AUI ユーザーは、画像や動画内の画像を見ることはできません。またドキュメントをスクリーンリーダーの読み上げによって聞いているため、音声コンテンツを視聴している時にはドキュメントの続きを読むことはできなくなります。

このことから、全てのドキュメントは、テキストを読むだけで説明することができ、画像や音などの情報はそれを補足するコンテンツとして考えて下さい。そうすることによって、どんなユーザにもわかりやすいドキュメントを提示できますし、補足した写真などが、なぜそこにあるのかの意味を明確に示すことができます。

まずは、プレーンなテキストだけのコンテンツを作って下さい。そして、そのテキストを読み返して、ユーザが指示を理解できることを確かめて下さい。

ユーザが指示を理解できるであろう、という確信を持つことができたら、画像や音声などの、補足となるマルチメディアコンテンツを追加して下さい。

本書は、全ての文書をまずテキストのみで制作しました。その後、補足となる画像や音声を設置し、事例として理解できるように構成しています。

### ◆ドキュメントの文字コードがわかるようにして下さい

日本語でドキュメントを制作する時には、文字コードの違いを配慮しなければなりません。記述してある文字コードと違う表示でユーザがドキュメントを閲覧してしまうと、文字が正しく表示されず、ユーザは困惑してしまうからです。

下記の表では、「ソフトウェア AUI ユーザビリティガイドライン」という文字を Shift_JIS、JIS_ISO2022、EUC_JP、UTF8 という 4 種類の文字コードで表記しています。SHIFT_JIS コードで記載したテキストは正しく表示されているはずですが、それ以外の文字コードで記述したテキストは理解できない内容になっているはずです。

**表 3『文字コードによる表記の違い』**

| 文字コード | 表記内容 |
| --- | --- |
| SHIFT_JIS | ソフトウェア AUI ユーザビリティガイドライン |
| JIS_ISO2022 | $B%c(Bt$B%H%&%'%"(B AUI$B%f!<%6%S%j%F%#%,%$%I%i%$%s(B |
| EUC_JP | ･縟･ﾈ･ｦ･ｧ･｢ AUI･譯ｼ･ｶ･ﾓ･･ﾆ･｣･ｬ･､･ﾉ･鬣､･[illegible] |
| UTF_8 | 繝｣t 繝医え繧ｧ繧｢ AUI 繝ｦ繝ｼ繧ｶ繝薙Μ繝・ぅ繧ｬ繧､繝峨Λ繧､繝ｳ |

日本の Windows の標準的な文字コードは SHIFT_JIS です。このため、特に SHIFT_JIS 以外の文字コードで記述したドキュメントを Windows のユーザが閲覧しようとすると、正しく表示できなくなることがあります。またスクリーンリーダーによっては、文字コードが違うために何も読み上げなくなることがあります。

このことから、ドキュメント内には文字コードを明記し、ユーザがそれに気づいて正しく表示できるようにすることが必要です。またこの記述をすることで、ドキュメントを表示するアプリケーションが文字コードを自動的に判断し、ユーザが意識しなくても正しい表示を行なえるようになります。

例えば HTML のドキュメントでは、文字コードを定義する HTML タグを記述して下さい。WEB ブラウザは、この文字コードの HTML タグを読み取って、適切な表示を行なっていますが、もし文字コードの記述が無いドキュメントがあると、正しく表示できなくなる可能性があります。

またテキスト文書では、ファイル名に「Readme_UTF8.txt」のように文字コード名を付記するか、ドキュメントの冒頭に*「This document charset is EUC_JP.」*などと記述します。英文で記述するのは、上記の表の通り、いずれの文字コードでも英文だけは正しく表示できるからです。

**◆説明書には、HTML を活用して下さい**

アプリケーションの取り扱い説明書などを作るための方法はさまざまですが、AUI ユーザーにとって読みやすいドキュメントを提供する時には、HTML を用いたドキュメントにすることをお薦めします。

HTML を使うことで、AUI ユーザーは最低限必要とするテキスト情報を容易に抽出することができます。また WEB アクセシビリティに配慮して HTML タグを適切に用いることで、テキストから特定の文書を検索したり、ドキュメントに何が書かれてあるのか、その全体像を短時間で把握できるようになります。

Windows には、この HTML を利用したヘルプフォーマットとして、『HTML ヘルプ(chm)』が設けられています。このヘルプは、通常の HTML と同様に AUI でアクセスし読み進めることができます。
ただし、AUI ユーザーの一部は、この HTML ヘルプの読み方を知らないことがあります。これは、HTML ヘルプはインターネットブラウザと比べた時に、キーボードでの操作が僅かに異なっていることと、その説明をスクリーンリーダーベンダーがユーザに対して、充分に提供していないからです。
このことから、HTML ヘルプの特長や利用方法についての説明を作成することをお薦めします。付録 4『HTML ヘルプにおけるキーボード操作』では、HTML ヘルプを AUI ユーザーが閲覧するための操作方法を紹介しています。

**◆PDF ドキュメントは、正常に読み上げができることを確認して下さい**

現在、電子文書を手軽に配布するための方法として、PDF が用いられています。PDF は、アドビ システムズ社が提唱したドキュメントの規格で、世界各国で利用されています。
この PDF は、AUI ユーザーでもスクリーンリーダーを通して読み取ることができます。また PDF を表示するアプリケーションの中には、PDF の内容を自動的に音声で読み上げる機能を搭載しているものもあります。

しかし、PDF 文書を読み上げるためには、ドキュメントが論理的に正しく構成されていなければなりません。不適切な PDF を読み上げようとすると、スクリーンリーダーがフリーズしてしまうからです。
論理的に正しく構成された PDF を作るためには、WEB ページと同じように PDF 文書に見出し、段落、表組などの***タグ***が設定されていることが必要です。
例えば WEB アクセシビリティに配慮した HTML ファイルのタグ情報を用いて PDF 文書を作成します。こうすると、WEB ページと同じ読み易さを持った PDF 文書を作ることができます。

もし、PDF を制作するアプリケーションにタグを付与する機能が搭載されていない時は、PDF 文書を閲覧できるソフトウェアを用いて、テキストデータを出力して下さい。そして、PDF に記載していた情報が過不足なくテキスト

に出力できているか、そのテキストを拡大・縮小しても読みやすさが損なわれないかどうかを確認して下さい。

**【 コラム 】 不適切な PDF を読み上げようとするとフリーズする原因**

不適切な PDF を読み上げようとすると、フリーズしてしまう原因は、スクリーンリーダーが PDF の文頭から文末までを一度に読み上げようとしてしまうためです。

スクリーンリーダーが文書を読み上げる時は、ある記号や文字を一つの区切りとして、そこまでの情報を読み上げます。例えばテキストデータをスクリーンリーダーが読み上げる時は、段落を区切る**改行文字**を区切りとします。

これに対して WEB ページや PDF での区切りは、見出し「<H1>」や、段落「<P>」といったタグと呼ばれる文字です。このタグで囲まれている内容には何らかの意味があるはずですから、一度に読み込むのが適切な判断です。

しかし、こうしたタグが文書中にどのくらいあるのか、といったことや、開始位置・終了位置の情報を自動的に調べるには、実際に文書を読み込まなければ知ることはできません。このため、こうしたタグの全く設定されていない PDF では、文頭から文末までの文字を一度に読み込んでしまうのです。

第６章で紹介した通り、音声読み上げを行なう処理は、音声で読み上げようとする文字が多いほど時間がかかります。だから全文を一度に音声読み上げさせようとすると、数分～数十分も処理に時間がかかってしまいます。

このことから、PDF 文書でもタグを用いてドキュメントが構造化されていることが必要です。また音声で読み上げるに当たって、文書が長文となる可能性がある時には、途中で処理を分割するなどの工夫が必要です。

**◆異なる章やページを参照する時に、ページ番号だけを記述しないで下さい**

取扱説明書や論文といったドキュメントでは、しばしば章やページを参照したり、他の文献を引用したりします。その時には、章題や文献名、ページ番号を記します。

この参照先の明記について、「詳細は 16 ページを参照」のようにページ番号だけを記すことは避けて下さい。ユーザが文献を拡大している場合や、テキストだけを抽出し読んでいる場合に、参照するページがずれてしまう可能性があるからです。

AUI ユーザーが本書を読むには、PDF 文書を入手しテキストを抽出することが最も確実です。しかしこの方法で抽出されるテキストにはページ番号が含まれていないか、含まれていてもそのページ番号がページの上にあるのか下にあるのかがわからないことがあります。

このため、参照先ページ番号を検索することは容易ではありません。本書では「第７章『ドキュメント』」といった記述に加え、リンクを張りクリックすることで参照できるようにしています。

ページ番号を記述する必要がある時には、「26ページ(第２章第３節)」のように、ページ名だけでなく参照する章や節番号を記述して下さい。こうすることで、ページ番号だけを探すよりもわかりやすくなります。

**【 コラム 】 本書に注釈が無い理由**

本書では、注釈を一切設けていません。その理由は、AUIユーザーが本書を読むに当たって、注釈を参照することが困難だからです。

ワードプロセッサなどで制作したドキュメントでは注釈を加えると、同じページ内に注釈欄が設けられ表示されます。しかしその文書にAUIでアクセスした場合、以下のいずれかの現象が起こります。

1. 注釈へはキーボードでアクセスできないので、注釈があることに気づかない
2. 注釈記号のある文面から注釈内のテキストへカーソルが移動する。しかし、元の位置に戻れなくなることがある。
3. 文書の末尾に注釈が付け加えられていると解釈される。この場合も注釈があることに気づかない。

上記の問題を解決するためには、注釈機能をサポートするアプリケーションが、キーボードで注釈部分をフォーカスした際にどのように対応するかを取り決め、ユーザへ向けて周知することが必要です。

例えば注釈のあるテキストをクリックすることで注釈部へフォーカスが移動し、キャンセル操作を行なうことで元の位置へ戻ってくる、といった仕組みを設けます。そして、この操作方法をドキュメントに解説します。

### ◆ドキュメントにはアウトラインを設けて下さい

アプリケーションソフトウェアの取り扱い説明書や電子文書といったドキュメントは通常、複数の章で構成されています。本書もそのドキュメントの一つです。

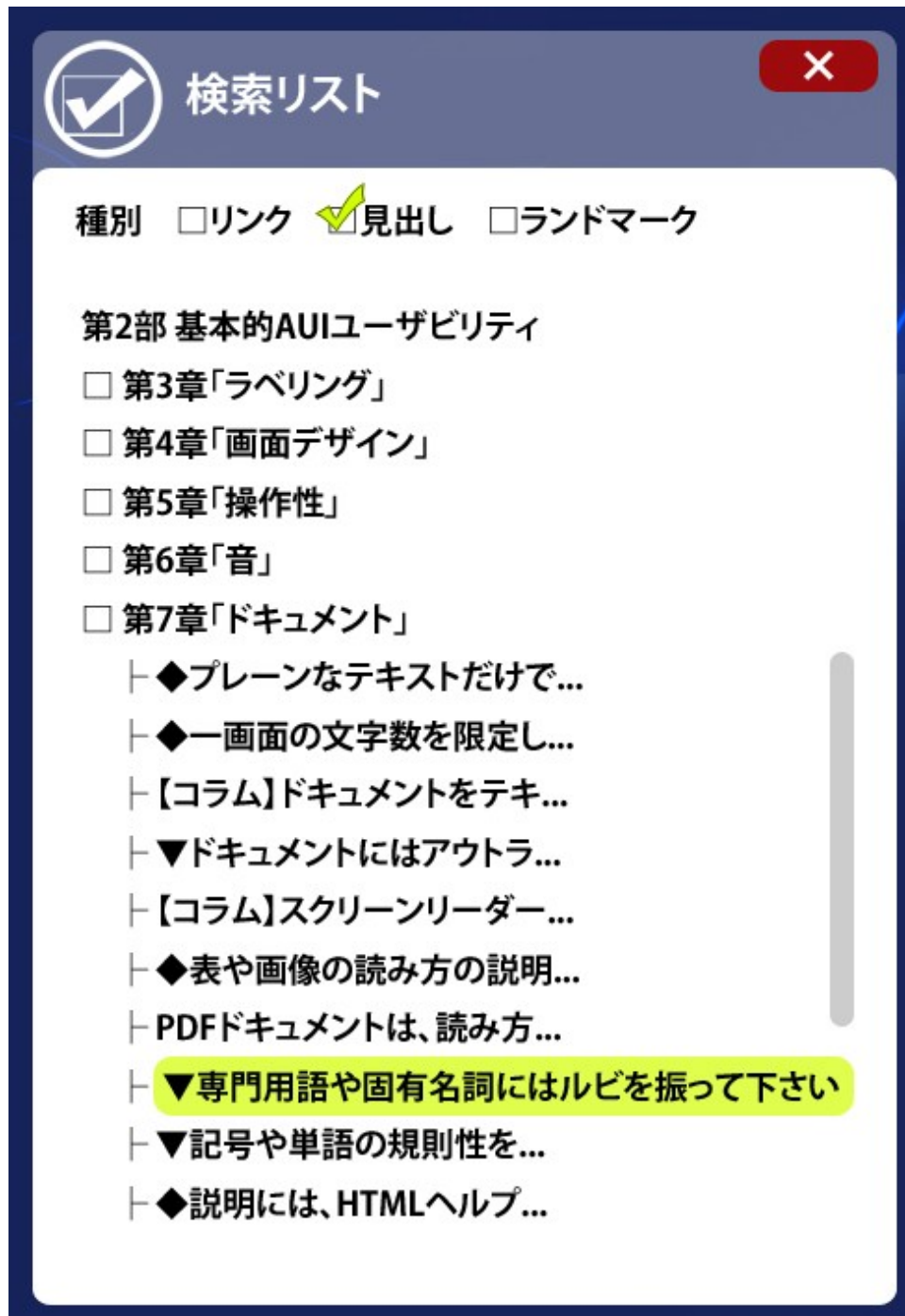
このような長文の文書には、何が書かれてあるのかを示す要約や目次といったアウトラインが必要です。またアウトラインは、文書の先頭部にあり、ユーザが文書を画面に表示してすぐに見つけられることが必要です。

特にAUIユーザーにとって、アウトラインがあることはとても重要です。AUIユーザーは画面に表示されたドキュメントを一望することができないため、現在何ページ目の何というコンテンツを読んでいるのか、またその前後のコンテンツが何であるのかを知るためには、実際にその部分を手動で選択し読み上げることが必要だからです。

またアウトラインを記述したセクション(アウトラインセクション)は、本文セクションなどのコンテンツと区別できることが必要です。せっかくアウトラインを設置しても、本文セクションと内容が似通っていると、ユーザは見落としてしまうからです。

例えばアウトラインの書かれてある部分を四角い枠で囲み、そのタイトルに**アウトライン**と明示すれば、誰もが「そのセクションがアウトラインである」と理解できます。このような区別が AUI ユーザーにもわかるようにすることが必要です。

現在、個々のセクションの区別を行なうのに最も適している方法は、各セクションに**見出し**を設けることです。見出しを付けたテキストは、下の画像のようにアウトラインとしてリストアップして表示され、AUI ユーザーはこのリストからコンテンツを検索し、読むことができます。

画像 11『見出しの一覧』

**【 コラム 】 スクリーンリーダーの課題**

アウトラインを含めた個々のセクションを区別するために枠線などで区切ることを紹介しましたが、残念ながら現在のスクリーンリーダーは、テキストと、「太字・斜体・見出し」といった一部の書式情報しか抽出しません。このため、セクションを区切るために枠線を引いても、AUI ユーザーがそれを見つけることはできません。

現在、AUI ユーザーがセクションを把握できる条件は、見出しの情報しかありません。このため、例えば、ヘッダーやフッターセクションに、あえて「ヘッダー」や「フッター」という文字を明記しなければ、AUI ユーザーは「ヘッダーセクションがある、フッターセクションがある」という情報を得られないからです。

このような問題に対応するためには、スクリーンリーダーが、文書の枠線や色情報を認識し、ユーザにフィードバックするように整備されなければなりません。付録 5『スクリーンリーダーで読み取れる情報』では、実際にスクリーンリーダーが AUI ユーザーに提供できている情報を知ることができます。

**◆表や画像の読み方の説明は、それよりも前に設けて下さい**

ドキュメントに表や画像を設ける場合、「下記の表は～～」や「下記の画像の～～」のような説明を追記することがあります。本書でも画像や表のあるセクションには、このような説明を付記しています。

この時の説明テキストは、原則として、該当する表や画像よりも前に記述されていることが必要です。多くのユーザは、ドキュメントを上から下、左から右に読むので、対象とする表や画像が説明よりも前にあると、先にそちらを読んでしまい、困惑するからです。

特に AUI ユーザーは、前後に書かれている内容が何であるのか、読み上げるまで把握できないので、「表や画像がある」ことや、「その説明がある」かどうかもわかりません。ですから、説明の次の段落に表や画像があることを明示する意味でも、必ず事前説明を設けて下さい。

**◆専門用語や固有名詞を用いる時は、それを解説するセクションを設けて下さい**

ドキュメントに専門用語や固有名詞を記述する時には、対象としているユーザがその用語を知っているかどうか、注意して下さい。ユーザは、知らない用語が多数出現すると、何から調べればいいのかがわからなくなり困惑してしまうからです。

この対策として、ドキュメント内に用語集を設けることが有効です。本書で登場する用語についても、付録 3『用語集』に紹介しています。

また用語の読み方が難しい時や、固有名詞である時にはルビを振ることも有効です。これはスクリーンリーダーがその用語を誤って読み上げたために、ユーザがその発音で正しいと勘違いをしてしまうことがあるからです。

**【 コラム 】 私はショクマです**

著者の苗字**諸熊**は、「モロクマ」と発音します。しかし多くのスクリーンリーダーは正しく発音できないため、同じ視覚障害を持つ方から「ショクマさん」などと呼ばれることがあります。

また「諸熊」を漢字変換すると、「諸隈」という同音の苗字が出てきますが、こちらはスクリーンリーダーの多くが正しく「モロクマ」と発音できます。このため、スクリーンリーダーの発音を信頼して苗字を書いた結果「諸隈さん」と書かれてしまうこともあります。

第２章で紹介した通り、AUI ユーザーは文字を発音の違いによって使い分けています。このため、固有名詞や、特殊な発音を利用した単語に対して、勘違いをしたり、間違えて覚えてしまうことがあります。

このことから、固有名詞を用いる際にはルビなどを割り振ることが有効です。AUI ユーザーは正しい発音を知ることで、それをスクリーンリーダーの読み上げに反映させることができるからです。

### ◆記号や単語の規則を守るようにして下さい

テキストでは、一部の文字を大文字にしたり、数字と漢数字を織り交ぜることで表現を目立たせることができます。しかしスクリーンリーダーを使うと、この表現があるために正しく読み上げることができない場合があります。

例えば、「１９８０年１０月２０日」と書くところを「１９８〇年１〇月２〇日」と書きます。スクリーンリーダーは「〇」は直前の数字とは関係がないと判断するため、「ヒャクキュウジュウハチ ゼロネン イチ ゼロガツ ニ ゼロニチ」と読み上げるか、記号を無視して「ヒャクキュウジュウハチ ネン イチ ガツ ニ ニチ」と読み上げてしまいます。

また「2011/10/20」と記述すると、スクリーンリーダーは「2010 ÷ 10 ÷ 20」という数式と解釈するか、「10 分の 2010 ÷ 20」という分数混在の表記と解釈してしまうことがあります。

このような誤った読み上げによる誤解を防ぐためには、単語や数値は正確に記述するようにして下さい。もしくは、正確な読み方を補足するルビを割り振るようにして下さい。

**表 4『表記を改善する事例』**

| 原文 | 修正事例 |
| --- | --- |
| 2011/10/20 | 2011 年 10 月 20 日、2011/10/20（2011年10月20日） |
| AUI ユーザビリティガイドライン | AUI ユーザビリティガイドライン（エーユーアイユーザビリティガイドライン） |

# 第３部　より深く AUI ユーザビリティを学ぶ

第１部では、AUI ユーザーの特長を、第２部では、具体的にアプリケーションに実装できる AUI ユーザー向けの配慮について紹介してきました。

第３部では、AUI ユーザーに最大限の使いやすさを提供することを念頭に置いています。具体的には、ユーザに提示する音声情報を操作する方法と、その際の留意点を紹介しています。

第８章では、提示する音声の順序と、最適な音声情報の選択方法を記述しています。

第９章では、直接音声読み上げさせる方法を挙げ、より充分な情報を提示する方法について記述しています。

第 10 章では、本書で紹介した観点を 10 の項目に集約しています。アプリケーションの AUI ユーザビリティを確認する時に、チェックするよう心がけていただきたいと思います。

## 第８章『音声化される情報をコントロールする』

本書ではこれまでに、**音声化される情報**や**順序**というキーワードを何度か挙げてきました。

AUI は、ユーザに提示する情報の順序を意識しなければなりません。それは**音**という情報は、伝達するために必ず時間がかかるだけでなく、一度の操作で一つの情報しか伝達できないためです。

例えば、多くの読者はこのガイドラインを画面に表示して、スクロール、またはページをめくりながら読み進めているはずです。ところが、これを AUI ユーザーが読む時には、現在フォーカスしている部分しか読み取りできず、しかも一文字ずつ音声を聞かなければいけません。

このため、AUI ユーザーは本書を読むのに多くの時間がかかります。また時間がかかる分、どんなことが書いてあったのかを覚えることも困難になります。

※具体的に※本書の文頭から文末(付録含む)までを、人が話す程度の速さで読み上げ続けると、約 2 時間 30 分要します。

そこで、音声化してユーザに提供する情報の内容や順序を制御することが有効です。例えば、最初に要約となる情報を提示し、次に詳細情報を提示することで、最初に提示された要約の情報が、強く記憶に残りやすくなります。

この章では、そのような課題に直面した時、どのように解決すればいいのか、その考え方を記述しています。

---

### ◆音声読み上げは、次の操作に移るためのヒントであることを意識して下さい

AUI は、音声を通してユーザに情報を提供しますが、ユーザは必ずしも、その情報の全てを聞くことが必要ではありません。

GUI ユーザーは、初めて遭遇した風景は、短い間注視するだけで、第一印象として記憶に残ります。これと同じように、AUI ユーザーも、初めて聞こえてきた音声の最初の何文字かを読んでいる内に、その音声が自身にとって、どのくらい重要な内容であるのかを判断します。

AUI ユーザーはパソコンを操作する時、スクリーンリーダーなどから提示された音声を聞きます。しかし、その音声を聞いて、現在の状況や次の操作をどうするべきかの判断ができたら、たとえ読み上げの途中でも次の操作に移りたいと考えます。

例えば、リスト内にある情報を次々と音声で提供する時には、ユーザが最も知りたがっている内容を最初に提供することが重要です。ユーザが知りたがっている内容というのは、それを聞いただけで、現在選択している情報源が正しいか、そうでないのか、次の操作に移ってもよいのかを知るためのサインのようなものです。

もしユーザが何を求めているのかが予測できないのならば、その項目を声に出してみて、最初の発音が異なる情報を優先するようにして下さい。例えば時刻や日付のような、初めの何文字かが同じものになる可能性のある項目を最初に読み上げてはいけません。
逆にユーザが求めている内容が明らかであるならば、その情報だけを抜粋するのは有効です。不必要な選択肢があるためにそれを聞く時間を浪費することを防げるからです。

例えば下のリストは、電子メールソフトで、「私からあなたに宛てて送信したメール」です。しかし、このメールをAUIユーザーが読むと、いつまでも用件がわからないことに苛々します。

**表5『受信メール一覧』**

| | | |
|---|---|---|
| 日時：2011/11/29 11:41 | 送信者：ショッピングHRT | 件名：本日のお薦め |
| 日時：2011/11/29 11:33 | 送信者：Morokuma hiroto | 件名：re^3:進捗状況 |
| 日時：2011/11/29 11:11 | 送信者：Morokuma hiroto | 件名：re:進捗 |
| 日時：2011/11/29 10:38 | 送信者：Morokuma hiroto | 件名：明日の予定について |
| 日時：2011/11/28 19:01 | 送信者：諸熊　浩人 | 件名：Re^5:複数の画面について |
| 日時：2011/11/28 16:23 | 送信者：諸熊　浩人 | 件名：Re^3:複数の画面について |
| 日時：2011/11/28 13:31 | 送信者：諸熊　浩人 | 件名：Re:複数の画面について |

ユーザが、「どんなメールが来たのか」を知りたいのであれば、上の並べ方ではいつまでも件名をしることができず、ストレスになります。
またユーザが、「28日に受信したメールを知りたい」としたとすると、「28」という数字を集中して聞こうとします。しかし、「23」や「29」などの、探したい数値に近い数値があるため、連続で数値を聞いていると困惑してしまいます。

そこで、上の表を以下のように並べ替えます。

| 表6『並び替えVer 受信メール一覧』 | | |
|---|---|---|
| 件名：本日のお薦め | 送信者：ショッピングHRT | 日時：2011/11/29 11:41 |
| 件名：re^3:進捗状況 | 送信者：Morokuma hiroto | 日時：2011/11/29 11:33 |
| 件名：re:進捗 | 送信者：Morokuma hiroto | 日時：2011/11/29 11:11 |
| 件名：明日の予定について | 送信者：Morokuma hiroto | 日時：2011/11/29 10:38 |
| 件名：Re^5:複数の画面について | 送信者：諸熊 浩人 | 日時：2011/11/28 19:01 |
| 件名：Re^3:複数の画面について | 送信者：諸熊 浩人 | 日時：2011/11/28 16:23 |
| 件名：Re:複数の画面について | 送信者：諸熊 浩人 | 日時：2011/11/28 13:31 |

最初に件名が設定されていることで、ユーザは興味を持っていないメールを受信しても、件名を読むだけで、それを捨てるかどうかをすぐに判断することができます。

また事前に、日時以外の文字情報を聞いていることで、数値を連続で聞くよりも集中力を保つことができます。さらに電子メールでのやり取りをユーザが知っていれば、件名から受信日時を推定することもできます。

※なお「件名」や「日時」などのキーとなる文字が最初に配置してあり、そのセルにどんなデータがあるのかをユーザが知っている時は、読み上げる必要はありません。ユーザがすでに知っている内容を読み上げると、その分の時間を浪費してしまうからです。

**◆一度に音声化される情報の量が多くなり過ぎないようにして下さい**

一般的なナレーションでは、一度に数百文字以上の音声を聞くこともありますが、その内容を長時間覚える必要はありません。これに対しAUIユーザーがパソコンを操作する時の音声は、作業を遂行するまでの間、できるだけ正確に覚えている必要があります。

このことから、AUIユーザーに一度に大量の音声を提示することは避けて下さい。例えば、一度に音声化されるキーワードを7つ前後にし、且つ20秒以内に読み終えることのできるようにする、といった規準を設けると良いかもしれません。

1956年に心理学者のジョージ・ミラーが発表した論文に、**『マジックナンバー』**と呼ばれる有名な定義があります。マジックナンバーは、**人は同時に7±2つの情報を記憶することができる**という内容で、しばしば電話番号の暗記の事例に用いられています。

さらに、情報を保持できる時間についても、短期記憶に関する研究で**20秒程度**という発表がされています。

このことから、聞き終えるまでに20秒以上かかり、多数のキーワードを含む音声情報は適量でないかもしれません。AUIユーザーが記憶するための負担が大きく、結局は忘れることで操作効率を低下させてしまう可能性があるからです。

もし、ユーザに上述の推奨値よりも長い音声を提供しなければならない時には、情報を分割して提供して下さい。

例えば分割したい情報に句読点を設置すると、スクリーンリーダーは句読点の位置まで読み上げ、次の文書をユーザが選択すれば続きを読み上げる、といったことができます。これによって、一度にユーザが聞く音声の情報量を少なくすることができます。

**◆音声の聞きなおしは、何度でもできるようにして下さい**

前項で述べた通り、AUIユーザーは、音を聞くのに時間を必要とするため、聞いた音を忘れてしまい、再度確認を必要とするかもしれません。

これは、音声を覚えやすいように情報を整理しても、必要なことです。その音声がユーザにとって適量であるかどうかは、ユーザの個人差だけでなく、もともとの文書の内容や、それを聞く場所の違いによる影響を受けるからです。

そこで音声は、ユーザの指定した位置から再度聞きなおしのできる手段を設けて下さい。またその聞き直しは、ユーザの判断で何度でも聞きなおしできることが重要です。

例えば、ユーザが本書のこの項を先頭から読み進めている時、直前の内容を聞きなおしたいと考えたら、「Ctrl+上矢印キー」を押します。すると、一つ前の段落から読み上げを再開し、ユーザは再び内容を聞くことができます。

## 第９章『音声読み上げを制御する』

第４章で紹介した通り、WindowsAPI 以外の画面デザインを用いたアプリケーションソフトウェアは、スクリーンリーダーで充分に読み取ることができないので、AUI ユーザーが使うことは困難です。

しかし、そのようなアプリケーションでも AUI ユーザーに使えるようにする方法があります。それは、直接音声読み上げを制御し、スクリーンリーダーと同等の情報を提示することです。

またこの方法を用いれば、WindowsAPI でもうまくユーザに伝えられない情報も伝達することもできます。例えばカーソルを移動させて項目の端に達したことや、文字の中に埋め込まれた見出し情報を伝える機能は WindowsAPI には存在しないので、それがユーザにとって意味がある情報ならば、利用する価値は充分にあります。

この章ではまず、音声読み上げを直接制御する方法を紹介します。その後、音声読み上げ機能を構築するに当たって留意する点を紹介します。

ただし、この方法を用いて既存のアプリケーションを AUI 対応にすることは推奨しません。

設計の段階で、AUI ユーザーへの配慮を取り入れていなかったアプリケーションを AUI で使えるようにするためには、専用機能の搭載が必要になります。具体的には、GUI ユーザーを想定したバージョンと、AUI ユーザーを想定したバージョンの二つを開発したり、アプリケーションを読み上げる仲介人のようなソフトウェアを配布しなければならなくなります。

こうなると、アプリケーションの開発後の管理は著しく不便になります。元となるアプリケーションや使用した音声化機能、例えばスクリーンリーダーなどがバージョンアップされる度に、制作した専用プログラムを更新しなければならなくなるためです。

---

### ◆音声制御の方法と特長

スクリーンリーダーで読み取ることのできない情報を音声化する方法は大きく分けて 3 つあります。可能であれば、全ての読み上げ方式をサポートし、個々のユーザが使いやすい方式を選択できるようにすることをお薦めします。

#### クリップボード

読み上げる情報をクリップボードに転送します。スクリーンリーダーがクリップボードを監視し読み上げる機能を活用する方法で、操作の説明を行なうのに適しています。

ただしクリップボードを使う操作はレスポンスが遅く、またクリップボード内のデータが常に上書きされるため、他の方法を搭載した上で、最後の選択肢として用いることをお薦めします。

**スクリーンリーダーの外部 DLL**

メーカーから提供されている API を使います。これを用いれば、ユーザは使いなれたスクリーンリーダーでアプリケーションが使用できます。ワードプロセッサや電子メールなど、文字情報を重視するアプリケーションで有効です。

全てのスクリーンリーダーに対応できるのであればこの機能は最も効果的ですが、それができない場合はユーザを限定してしまう点に注意して下さい。

**音声合成エンジンを操作**

Windows やメーカーから提供されている音声読み上げ API を利用し、直接音声を出します。文書の読み上げや操作説明に音声を使うアプリケーションで有効です。

この方法は 3 種類の中でも最も簡単に実装することができるため、多くのアプリケーションで用いられています。ただし日本語での読み上げをサポートすることは困難です。この詳細は後述します。

---

### ◆音声読み上げの声質や速度、音量などはユーザが変更できるようにして下さい

音声読み上げの声質や速度、または音量といった内容は、適切な初期値と共に、ユーザが自在に変更できることが必要です。なぜならば、ユーザが聞きやすい音声読み上げは、ユーザの個人差、使う場所、音声の聞き方などによって変化するからです。

特に、音声合成エンジンを直接操作して音声読み上げを提供しているアプリケーションでは、この機能を必ず持たせて下さい。この方法で提供する音声はスクリーンリーダーの機能ではないので、スクリーンリーダー側の設定を変更しても反映されないからです。

**音声の声質**

音声の声質というのは、男性の声・女性の声、といった声の種類や、声の高さなどを意味する属性です。ユーザや聞く場所によって、最適な声質は異なってきます。

例えば濁りのある声は、耳への負担が少なく長時間聞き続けることに適しているかもしれませんが、空調やファンなどの環境音にかき消されやすい、という問題点があります。また、音声合成エンジンの声によく似た人が近く

で話をしているような時には、別の声にしなければどちらの声がスクリーンリーダーの音声なのかがわかりにくくなってしまいます。

声質の初期値は、最も発音が明瞭で、声に濁りの少ないものを選んで下さい。初めてアプリケーションを使うユーザは、まず画面毎の音声ガイダンスの概要を覚えたいと考えるので、聞き違いを少なくできることが必要だからです。

声の高さの初期値は、中間値を選んで下さい。一般的な音声読み上げエンジンは中間値の高さが、ユーザにとって最も聞きやすい声になるように設計されているからです。

**音声の速度**

音声の速度とは、ゆっくりと読み上げるか早口で読み上げるか、という属性です。ゆっくり読み上げるほど内容の聞き取りは容易になりますが、その分時間がかかってしまいます。また音声合成エンジンによっては、ゆっくり読み上げると声が濁ってしまい、聞き取りにくくなるものがあります。

AUI ユーザーは、操作に熟練するほど音声の速度が速くなる傾向があります。ユーザは音声を聞かなければ次の操作に移れないため、音声の速度が速いほど、操作完了に要する時間を短くできるからです。

速度の初期値は、150 文字/分、または中間値(50%～60%)を採用して下さい。この値は、人がゆっくり発話する速さに近く、初心者のユーザがアプリケーションを音声ガイドを頼りに操作する時にも有効です。

**音声の音量**

全ての音声読み上げは、パソコンのスピーカーから再生されているとは限りません。音声の音量は、どこで読み上げを聞いているのかによって変化します。

例えば著者の場合、職場では、空調やプリンターの音が大きいことや、スピーカーから音声を出すと周囲の方に迷惑がかかることから、常にイヤホンを着用し音声を聞いています。

一方、家の自室では、音声を出しても誰も迷惑をかけないこと、職場でイヤホンを着け続け、耳を酷使するので、パソコンのスピーカーから音声を聞いています。

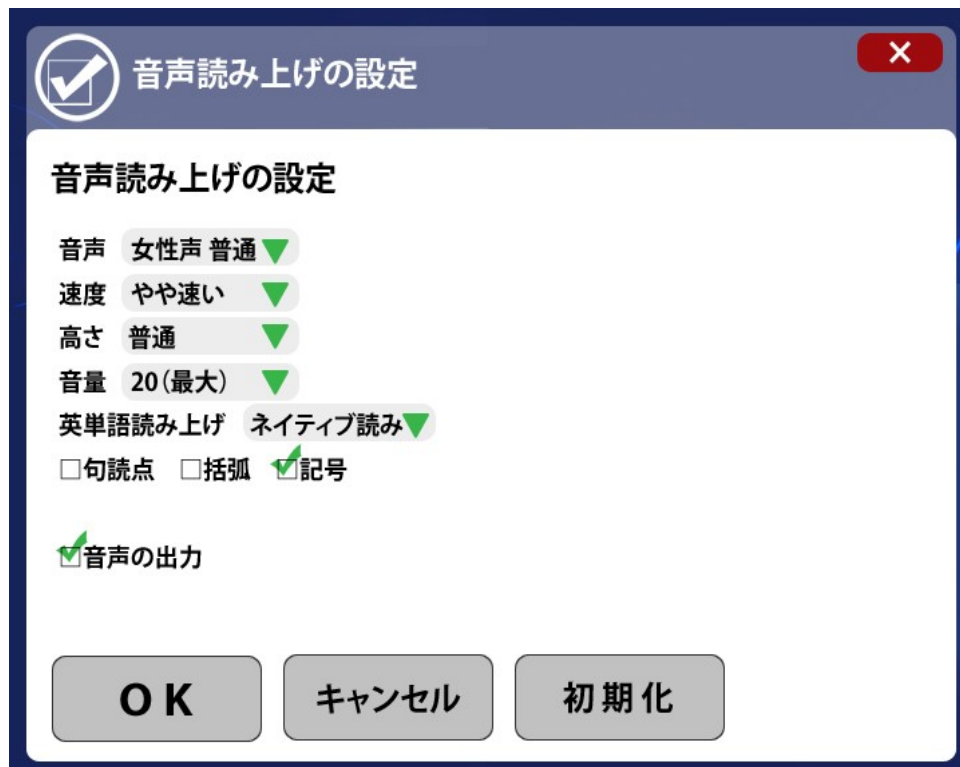
このように、音声を聞く場所、スピーカーなのかイヤホンなのか、という音声の出し方の違いにより、聞きやすい音量は頻繁に変わります。またスピーカーで聞くと音量が小さいが、イヤホンを付けると音量が大きく聞こえる場合もあります。

音量の初期値は、その音量がパソコンのスピーカー音量に連動しないのならば、最大にして下さい。こうすることによって、聞こえる音声の音量は、ユーザが設定しているスピーカーの音量とほぼ同じ程度になります。

一方で、音量を設定すると連動してパソコンのスピーカー音量が変わってしまう場合は、スピーカーの音量設定に従って下さい。そして、ユーザが音量を調節しない限り、絶対に変更しないようにして下さい。

**音声読み上げの設定の事例**

下の画像は、音声読み上げの設定を行なうフォームの例です。音声の声質や速さ、音量、そして声の高低を自在に設定することができます。

画像 12『音声の声質・速度などの設定画面』

**◆Speach API を使う時には、最初に日本語を読めることを確認して下さい**

独自に音声読み上げを利用する手段の一つとして、Windows には『Speach API』という機能が搭載されています。この API を用いることで、スクリーンリーダーが無くても音声読み上げを行なうことができます。

ただし、この『Speach API』を用いた日本語対応のアプリケーションを AUI ユーザーに提供する時には、ユーザのパソコンに日本語で読み上げることのできる音声合成エンジンが搭載されているかどうか、確認するようにして下さい。

音声合成エンジンの中には、日本語を読み上げることができない製品があります。このような音声合成エンジンで日本語のアプリケーションを読み上げさせると、ASCII コード以外の文字(全角の英数字も含む)を一切読み上げません。このため、日本の AUI ユーザーは読み上げ内容を理解することができず、全く使えなくなってしまいます。

この問題への具体的な対策は三つあります。同時に複数の手段を採用してもかまいません。

一つめは、アプリケーションのパッケージに日本語で読み上げられる音声合成エンジンを同梱し、同時に使用する方法です。音声合成エンジンを同時に配布しなければなりませんが、ユーザに導入の負担を与えることなく、日本語でアプリケーションを読み上げさせることができます。

二つめは、アプリケーションに**日本語が読み上げられる音声合成エンジンを自動的に検索する**機能を盛り込む方法です。もし存在しなかったら、別の方法、例えばクリップボードに読み上げ内容を転送するように設定します。

三つめは、アプリケーションの説明書などに、日本語で音声読み上げのできる音声合成エンジンの入手方法を記載し、ユーザにダウンロードしてもらう方法です。この方法は、多言語対応の製品で、専用の言語設定をユーザに選んでもらうことと似ています。

**【 コラム 】 日英混在文書の対応**

英語の音声合成エンジンでは日本語は一切読み上げないと記しました。これに対して、日本語の音声合成エンジンは英語を読み上げることができます。

ただし、多くの日本語の音声合成エンジンは英語に対して、日本語英語、またはフルスペルで読み上げるため、本来の英単語と違う意味でユーザに伝わってしまったり、英単語として読み上げないために困惑する可能性があります。

この問題に対応する一つの方法は、英単語を自動的に検知してそこだけを英語の音声合成エンジンに読み上げさせるようにすることです。英語の音声合成エンジンであれば、少なくとも日本語の音声合成エンジンより英単語を正しく読み上げるので、ユーザに誤解させる可能性を低くできるからです。

ただしこの方法を使用した場合、読み上げの途中で音声合成エンジンが切り替わるため、一瞬違う声になりユーザが困惑する可能性があります。例えば男性の声で読み上げていたのが、英単語だけ女性の声で読み上げられることで、その部分に特に重要な意味がなくても強調されてしまいます。

また、項目名やページ番号で用いられる「A」や「I」、WEB ページの URL のように、スペル事態に大きな意味がある情報を読み上げる時には、日本の

音声合成エンジンがフルスペルで読み上げた方がわかりやすくなるかもしれません。

**◆全文読み以外の読み上げ方式をサポートして下さい**

音声読み上げ機能を独自に提供する時には、第8章で述べた「ユーザが覚えやすい音声情報を作る」ことが特に重要です。独自の音声読み上げ機能を利用しているということは、その他の方法を用いてユーザがアプリケーションを操作することが困難だからです。

具体的には、読み上げの方式として全文の読み上げ以外に、選択しているセクション・段落・文節・単語・文字などの単位で読み上げができるようにして下さい。特に段落・文節・文字単位で読み上げを行なうことは、ワードプロセッサやインターネットブラウザのような、テキスト情報が大きな役割を持つアプリケーションでは必須です。

また文字単位で読み上げる時には、句読点や記号、空白や改行文字などが、ユーザに通知されるようにして下さい。カーソルを動かしているのに無音であると、ユーザはそこに何が書いてあるのかがわからないからです。

なお音声合成エンジンの多くは、句読点や記号、空白文字（くうはくもじ）は読み上げることはできません。これらの文字を読み上げる時には、該当文字を「トーテン」、「スペース」、「カイギョー」のような読み仮名に置き変えることを推奨します。

**◆読み上げの途中に新しい読み上げが生じた時の対応を検討して下さい**

AUI ユーザー向けのアプリケーションでは、直前の内容を読み上げている最中に、新しい文字の読み上げが必要になることがあります。

例えばホームページを読み上げている途中に、新しい記事が追加された場合、新しい記事に対してどのように対応するか、またある内容を読み上げている途中にユーザが次の内容を知りたいと考えて操作を行なった時、途中まで読み上げていた情報はどうするのか、といったことを検討する必要があります。

音声読み上げの最中に別の音声読み上げが生じた時には、以下の4つの手段からどれか一つを選択します。適切な選択をするためには、アプリケーションの目的や、読み上げが重なった時の状況を分析して判断します。

1. 新しい内容の読み上げは行わず、既存の読み上げは続ける。その代わりに情報が更新されたことだけを効果音などで通知する。

2. 現在読み上げている内容が終了するのを待ち、その後に新しい内容の読み上げを開始する。
3. 現在読み上げている内容を中断し、新しい内容を読み上げる。
4. 現在読み上げている内容の音声に重ねる形で、新しい内容の読み上げを開始する。

例えば、アプリケーションではリアルタイム性が重視されており新しい情報を素早く読み上げることが優先される場合には、3 や 4 のような選択を行ないます。またスクリーンリーダーのように、ユーザの操作に対して音声読み上げを提供するアプリケーションは、キー操作に対してすぐにフィードバックすることが必要なので、3 を選択します。

なお 4 の方式で、直前の音声と新しい音声を同時に再生する時には、全く違う声にすることや、再生される音声の位置をずらすようにして下さい。こうすることで多数の音声の中から選択した音声だけを集中して聞くことができる性質**カクテルパーティ効果**が成立します。
ただし、カクテルパーティ効果を使っていいのは、同時に複数の情報を提示する必要がある時だけです。単にキー操作に対する情報の提示としてこの用法を用いると、いつまでも音声が止まらずうるさくなってしまいます。

**◆音声読み上げは単調にならないようにして下さい**

前項までに、ユーザが聞きやすいように音声を調節できることが重要である、と述べてきました。しかし、合成音声は単調なので、長時間聞き続けると疲れてしまったり、重要な情報の読み上げが途中に入っていても、それを聞き流してしまう可能性があります。
このような時には、第 6 章で紹介した、情報の変化を音の変化で表現することが有効です。例えば見出し部を読み上げる時にはビープ音を鳴らすことや、大文字と小文字を読み上げる時に音声の高低を変更する、といった方法が活用されています。
もちろん、これは文字単位で読み上げる時にも有効です。例えば英語の大文字と小文字の違いを声の高さで区別すれば、英語に対して「大文字 A」、「小文字 a」などと読み上げさせる必要がなくなり、ユーザにもわかりやすくなります。

**◆ユーザがキーを押した時には必ず何かのフィードバックをして下さい**

AUI ユーザー向けのアプリケーションでは、取扱説明書等に記したキーをユーザが押した時には、いかなる場合でも音によるフィードバックを行なうことが必要です。キーを押したのに反応がないと、アプリケーションがフリーズしたと勘違いをしてしまうからです。

またアプリケーションが処理中などでキー操作を受け付けられない時は、処理中であることを効果音などを用いて定期的に通知して下さい。効果音が鳴っていることで、正常にアプリケーションが動作していることを知ることができますし、それが鳴り終わった時には「処理が完了した」ということを推測できるからです。

もし、AUI によるフィードバックのタイミングが把握できない時には、アプリケーション全体を通して、AUI によるフィードバックが得られない時間が 5 秒以上続いている部分をなくす、といった規準を設けて下さい。
ただしこの時間には、音声読み上げの時間は含めないで下さい。音声読み上げが終了する前に次のフィードバックを行なうことで、AUI ユーザーが音声読み上げを聞き取れなくなる可能性があるからです。

例えば、AUI ユーザーがリストボックスを操作する時、リスト項目が一つしかないためにカーソルが移動しなかった時や、項目が一つも無いために何もフォーカスが変化しなかった時、操作前と操作後で情報の変化はありません。しかし操作したことへのフィードバックを怠ると、ユーザはなぜ操作を受け付けなかったのかに困惑してしまいます。
このような場合、項目が一つしか存在しない時には、現在選択中の内容を再度読み上げたり、ユーザの矢印キーの操作に対してカーソルが移動しなかったことを音声や効果音で通知したりすることが有効です。一方、何もフォーカスする対象が無い時には、項目が存在しないことを読み上げたり、効果音で通知したりすることが有効です。

**【 コラム 】 AUI ユーザーは矢印キーを好む？**

キーを押した時にフィードバックが必要なキーは、取扱説明書などに記したキーの他に、エンターキーや矢印キーなど、使用頻度の高いキーに設けることが有効です。特に矢印キーでの操作でフィードバックが得られることは、AUI ユーザーに喜ばれます。

AUI では、画面上の項目を探索するための基本的な操作は、矢印キーでカーソルを上下左右に移動させることです。このため、ユーザはこのキーを、他のキーと比べられないほど多く押します。
その操作が定着した結果、多くのユーザは現在何が選ばれているのかを知るためにさえ、まず矢印キーを押してカーソルを移動し、その後反対方向の矢印キーを押して、元の位置に戻る操作を行なうようになりました。
これは一見無駄な操作に思われます。しかし、現在選択中の項目を探すショートカットキーを押すよりシンプルで、確実です。ショートカットキーは

スクリーンリーダー毎に異なる上、両手を動かさなければならないことが多いからです。

これに対して矢印キーでの操作は、片手で実行できる上に、スクリーンリーダー毎の違いにも影響されません。このため、ショートカットキーを押すより、まず矢印キーを押してみて、もし何も読まなかった時初めて、該当のショートカットキーを押すことが生じます。

## 第 10 章『AUI10 の原則』

最後の章では、本書の総まとめとして、**AUI ユーザビリティの 10 原則**を紹介します。

この 10 原則は、本書を制作するに当たり、著者自身が考案したものです。これまでに行なってきた研究や調査の結果を集約し、心がける原則となることを目指しました。

アプリケーションがこの原則に適しているかどうかをチェックすることで、AUI ユーザーにとって使いやすく設計されているかどうかを確かめることができます。

---

**☆AUI10 原則**


1. 全てのデータにラベルを付けて下さい
2. アプリケーションの全ての機能にアクセスできるようにして下さい
3. デザインや API は利用する目的に沿った機能を優先して下さい
4. 情報の提示はテキストに主眼を置いて下さい
5. 音声読み上げや効果音は、ユーザがカスタマイズできるようにして下さい
6. 音声読み上げによるユーザへの記憶や探索の負荷を軽減して下さい
7. 動作の経過や結果は音でフィードバックして下さい
8. 時間制限を設ける時には、音を聞くための時間を考慮して下さい
9. フォーカスの制御権は常にユーザが持つようにして下さい
10. よく使う機能は、簡便にアクセスできるようにして下さい


---

### その１　全てのデータにラベルを付けて下さい

**◆チェック項目**

1. アプリケーションの全てのウィンドウにラベルが設定されているか？
2. ラベルは、個々のウィンドウの働きを示しているか？
3. ラベル名は過度に長いものでないか？

**◆説明**

ユーザは、表示されたウィンドウから、まず何ができるのか？ということを最も知りたいと考えます。この最大のヒントが、ウィンドウに設定されたラベルです。

全てのウィンドウやボタンに、それに対応したラベルを設定することで、作業の目的に迷ってしまうことを防ぐことができます。

ただし、過度に長いラベル名を付ける時には、その名称が似通ったラベルが無いかどうかを確かめて下さい。ユーザにとって重要なのは、ラベルがあることではなく、ラベル名から、現在どのウィンドウやボタンにアクセスし操作しているのか？を把握できることだからです。

### その２　アプリケーションの全ての機能にアクセスできるようにして下さい

**◆チェック項目**

1．全ての機能はマウス・キーボードのいずれかだけで利用できるか？
2．アクセスしたウィンドウからの情報は、スクリーンリーダーや効果音を通して読み取ることができるか？

**◆説明**

全ての機能にアクセスし情報を受け取れることは、現在の AUI ユーザーにとってとても喜ばしいことです。なぜならば、AUI ユーザーが利用している市販のアプリケーションソフトウェアのほとんどで、アクセスできない機能があるからです。

従って、まずは AUI ユーザーに利用できない機能を無くすことから始めることが必要です。これはユーザビリティとしてはとても基本的なことですが、達成しなければユーザビリティの向上は見込めません。

### その３　デザインや API は利用する目的に沿った機能を優先して下さい

**◆チェック項目**

1．デザインした画面を利用する目的は、一連の操作ステップに繋がるものであるか？
2．大量のウィンドウやボタンをただ列挙していないか？
3．デザインは一貫しているか？

**◆説明**

第４章で紹介した通り、ユーザはデザインした画面を見るためにアプリケーションを使っているわけではありません。ですから、不要な画面が多いほど、それを操作するための余分な時間・ストレスをユーザに与えてしまう点に注意しなければなりません。

またユーザにとって、画面デザインが一貫していることや、見慣れたものであることは、操作方法を覚える必要がなく、負担が少ないことを意味しています。

### その４　情報の提示はテキストに主眼を置いて下さい

**◆チェック項目**

1．テキストだけを読んで情報の意味は理解できるか？
2．設置した画像などは、テキストの補足として準備されたものであるか？

**◆説明**

　AUI ユーザーにとって、テキストの情報はとても大切です。なぜならば、現在のスクリーンリーダーは、テキスト以外の情報をほとんど読み取れないからです。

　しかし、テキストに主眼を置くことは、大多数のユーザにとっても必要なことです。画像や音、色の情報は、それを受け取るユーザによって解釈が曖昧になりますが、言語であるテキスト情報は、それを明確に示すことができるからです。

### その５　音声読み上げや効果音は、ユーザがカスタマイズできるようにして下さい

**◆チェック項目**

1. 音声の声質・速さ・音量などはユーザが変更できるか？
2. 音声読み上げが単調になっていないか
3. 重要な情報には効果音を付与しているか？
4. スクリーンリーダーを妨害する効果音はないか？

**◆説明**

　AUI を用いたアプリケーションは、必然的に同じ音声や効果音を何度も聞いたり、1 時間以上音声を聞き続けることが生じます。また GUI ユーザーと違う点として、パソコンを操作する場所の違い、例えば雑踏音などの影響を受けます。

　このことから、聞きやすい音声や効果音を選択できることがとても大切です。それも、ユーザが変更したいと思った時、いつでも変更できることが必要です。

### その６　音声読み上げによるユーザへの記憶や探索の負荷を軽減して下さい

**◆チェック項目**

1. 聞こえてきた音声は暗記しやすいか？
2. 音声はユーザが任意で繰り返し聞きなおすことができるか？
3. 項目名の先頭何文字かが似通ったものを列挙していないか？
4. 目的の作業を達成するための操作過程に、明らかに読み上げる必要のない内容が含まれていないか？

**◆説明**

　第８章で述べた通り、AUI ユーザーにとっての音声や効果音は、操作を遂行するためのヒントです。従って、その情報を素早く、そして正確に取得できることが重要です。

このことから、余分な音声ガイダンスを減らしたり、聞いていても似通っている音声読み上げを少なくすることは、とても大切です。

### その７　動作の経過や結果は音でフィードバックして下さい

**◆チェック項目**

1. 無音でフィードバックされる処理は無いか？
2. 意味もなく効果音を再生させていないか？
3. 再生された音は、何と何の違いを伝える要素であるのかがわかるものであるか？

**◆説明**

AUI は、音のフィードバックを受けて初めて次の操作に移るユーザインタフェースです。だから、何かの結果を必ず音でフィードバックすることと、フィードバックと関係の無い用法で音を使わないようにすることは、AUI の要となる考え方です。

まずはアプリケーションの全ての操作で、無音でフィードバックされる情報が無いかどうかを確かめて下さい。また同時に、フィードバックと関係の無い用法で音が用いられていないかどうかを確かめて下さい。

次のステップとして、フィードバックされる音と、フィードバックされる情報を結び付けて下さい。この音と情報との結びつきは、エラーメッセージと警告メッセージの表示が違うように、1 対 1 であることが望ましいです。

### その８　時間制限を設ける時には、音を聞くための時間を考慮して下さい

**◆チェック項目**

1. 操作に制限時間が設けられている場合、そのことを事前に説明しているか？
2. 事前説明や注意事項は時間をかけてゆっくり読めるようになっているか？
3. 操作に対する制限時間を超過した場合のやり直しはできるか？
4. 制限時間は、延長・解除することができるか？

**◆説明**

AUI は、操作に時間がかかるユーザインタフェースですから、GUI 環境と同じ制限時間内で操作を完了することはほとんど不可能です。しかも、制限時間が設けられていることを知るために、その制限時間内でメッセージを読まなければならないので、実際に指示通りに操作ができる時間はほとんどありません。

そこでまず必要なことは、制限時間が設けられているセクションに入る前の段階で、次のセクションで制限時間が設けられていることを説明することです。またこの時、具体的な操作手順や効率的な操作方法も説明すれば、操作方法を探索するための時間も省略することができます。

次に必要なことは、制限時間を超過した場合にやり直しができることです。制限時間の延長や解除も有効な方法ですが、それを実行するためのボタンをユーザが見つけられる、という保証がないからです。

## その９　フォーカスの制御権は常にユーザが持つようにして下さい

**◆チェック項目**

1. 自動的にフォーカスを移動させる処理が含まれていないか？
2. 階層構造の操作では、直前のフォーカス位置を記憶しているか？
3. カーソルを移動するキーとそれを押した時のカーソルの移動位置と量は常に一定であるか？
4. フォーカスを奪うようなポップアップをしていないか

**◆説明**

AUI ユーザーは画面を一望できるわけではないので、現在画面のどの辺りを操作しているのかを把握することは困難です。そして、その頼りとなる情報は、キーボードでフォーカスした位置と、そこから得られる情報です。

このことから、ユーザの意思に反して勝手にフォーカスを移動させたり失わせることは一切行なってはいけません。フォーカスを失っていい時というのは、初めて表示したウィンドウの時と、操作前と操作後で、表示されている情報のほぼ全てが違う時の 2 つだけです。

また、フォーカスしたからと言って、アクセスしたと見なしてはいけません。

スクリーンリーダーは、そこへフォーカスを移動させなければ情報を受け取ることができません。このため、「有るかどうか？」を調べたいだけなのに、アクセスとして反応してしまうと、ユーザは困惑してしまいます。

最後に、カーソルを移動させる操作と、操作の結果カーソルが移動する量は常に一定であることが必要です。この一貫性を崩すと、元の状態に戻せなくなるからです。

## その 10 よく使う機能は、簡便にアクセスできるようにして下さい

### ◆チェック項目

1．よく使う機能にはショートカットキーが割り当てられているか？
2．連続して同じキーを押す操作は省略する方法が設けられているか？
3．Windows 共通のショートカットキーに他の機能を割り当てていないか？
4．複数階層ある項目はメニュー化、またはグループ化されているか？

### ◆説明

AUI ユーザーが目的の機能を素早く、そして簡単に操作するための方法は、ショートカット機能の活用です。ショートカットキーを活用したり、項目間移動を素早く行なうキーを設けることはとても有効です。

また複数項目をグループ化し、メニューとすることで、目的の機能をカテゴリから選んで探すことができます。これはショートカットキーのような素早い操作は期待できませんが、必要な機能をじっくり探すことができるので、初心者のユーザには何よりも有効な手段です。

## 参考文献

### ◆ガイドライン

*1. 『Microsoft アクセシビリティ機能を備えたアプリケーションの設計』*
http://msdn.microsoft.com/ja-jp/library/cc421540.aspx

*2. 『アップル・ヒューマンインタフェースガイドライン』*
http://developer.apple.com/library/mac/navigation/index.html#topic=Guides§ion=Resource+Types

*3. 『Just Ask: デザインプロセスを通じて取り組むアクセシビリティ』*
http://www.uiaccess.com/justask/ja/

*4. 『JIS X 8341「高齢者・障害者等配慮設計指針－情報通信における機器，ソフトウェア及びサービス－」』*
http://www.jsa.or.jp/stdz/instac/commitee-acc/WG3/Guidex8341.html


### ◆論文

*1. 『視覚障害者のための電子メール環境における操作性の検討』*
西本 卓也，住吉 悠希，荒木 雅弘，新美 康永，ヒューマンインタフェース学会研究報告集 Vol.2 No.5, pp.33-38, Dec 2000.
http://hil.t.u-tokyo.ac.jp/~nishi/2000/Nishimoto2000HIS12/

*2. [PDF] 『SUI サイン音を用いた情報表示とそのデザイン』*
和氣 早苗，ヒューマンインタフェースシンポジウム 2005, 2005.
http://www-im.dwc.doshisha.ac.jp/~wake-lab/member/swake/pdf/SS-SUI-HIS05.pdf

*3. 音声インタフェースと Web アクセシビリティ*
西本卓也，第 4 回 SIGACI 研究談話会 / 第 30 回 UAI 研究会，Nov 2011.
http://www.slideshare.net/nishimotz/web-2316759

*4. 『理想のスクリーンリーダーはソフトウエア間対話の実現により可能となる』*
石川 准，WSIS フェーズ 2 サイドイベント：障害者コーカス主催「第 2 回情報社会における障害者のグローバル・フォーラム」，Nom 2005.
http://www.dinf.ne.jp/doc/japanese/prompt/ws051115_ishikawa_s.html

# 付録資料

## 付録１『スクリーンリーダー情報』

この資料は、2012 年現在、日本で普及しているスクリーンリーダーの製品情報やその特長を紹介しています。なお、より詳しい内容については、掲載している URL より開発メーカーのホームページを確認して下さい。

### ① PC-Talker・VDM シリーズ

メーカー：高地システム開発・アクセステクノロジー
http://www.pctalker.net/
http://accesstechnology.co.jp/

PC-Talker は、高知システム開発によって開発されたスクリーンリーダーで、現在の日本市場最も高いシェアを持っています。

PC-Talker の最も優れている特長は、音声の発音がはっきりしており、声に濁りがなく、そしてレスポンスが速いことです。この特長から、初心者から上級者まで多くの AUI ユーザーに活用されています。

VDM シリーズは、PC-Talker の操作性を MS-DOS の規準に合わせてカスタマイズしたスクリーンリーダーです。読み上げ方や性能に差異はほとんどないので、PC-Talker と統合して呼ばれることもあります。

### ② 95Reader・XPReader

メーカー：システムソリューションセンター栃木
http://www.ssct.co.jp/

XPReader は、システムソリューションセンター栃木によって開発された製品ですが、現在は開発を終了しています。視覚障害ユーザーの間では WindowsXP への人気が根強く、現在でも利用しているユーザが多くいます。

XPReader は、就労支援を目的に開発されたスクリーンリーダーなので、Microsoft Office 製品への対応に重点が置かれています。特に現代のプレゼンテーションの必需品となっている PowerPoint への対応は充実しており、特定スライドのスムーズな切り替えや、何分プレゼンテーションを行なったかなどの時間を計測する機能が設けられています。

また XPReader は、音声と効果音が重なった場合に、自動的に効果音の音量を低くする機能があります。これによって、大音量で音楽が鳴るような状況でも音声を頼りに操作することができます。

### ③ FocusTalk

メーカー：SkyFish
http://www.skyfish.co.jp/

FocusTalk は、SkyFish によって開発された製品です。XPReader の開発メンバーが参加しており、同製品のノウハウを後継しています。
スクリーンリーダーとしては XPReader と PC-Talker の中間に当たる特長を持っています。PC-Talker 向けのソフトウェアを利用することもでき、同時に XPReader に近い操作性を持っているので、幅広い製品を操作することができます。
また、ナレーション向けの音声合成エンジンが搭載されているので、とても音質が優れていることも特長です。それだけでなく、レスポンスを高めるための工夫もこなしてあるので、AUI において重大なレスポンス速度を維持しています。

### ④ Jaws(Job Access for Window System)

メーカー：Freedom Scientific(国内販売代理エクストラ)
http://www.extra.co.jp/

Jaws は、米国 Freedom Scientific 社によって開発されたスクリーンリーダーで、世界で最も広いシェアを持っています。海外でも運用するアプリケーションを開発する時には、Jaws で読み上げができるように対応することをお薦めします。
Jaws は就労に主眼を置いて開発された製品ですので、現在利用されている業務システム、業務用アプリケーションのほとんどに対応しています。現在、Microsoft Access や Visual C++ 統合開発環境などを使う手段は、Jaws を用いることのみです。
また Jaws は、独自のスクリプト言語を内蔵しており、ユーザがスクリプトを制作することができます。このスクリプトを用いれば、画面デザインに一切影響されず、AUI ユーザーに音声情報や独自のキーボード操作を提供することができます。

### ⑤ NVDA(NonVisual Desktop Access)

メーカー：オープンソース
http://sourceforge.jp/projects/nvdajp/

NVDA は、オープンソースで開発されているスクリーンリーダーです。現在、世界各国でローカライズと修正、そして意見交換も盛んに行われています。

NVDA は、無料で利用できますが、優れた読み上げ機能を持っています。

例えばワードプロセッサでの読み上げでは、文字フォントが通常と異なる文字を探すため、それまでは一文字ずつ文字を選択しては**詳細読み**というショートカットキーを押さなければなりませんでした。が、NVDA では、ただ文頭から読ませ続けるだけで、通常と異なるフォントの文字を見つけると、それを音声ガイダンスに含めて読み上げてくれます。

現在の NVDA の課題は、全ての音声情報が単調な音声で読み上げられていることです。このため、音声を聞き続けていると、情報の違いがわかりにくくなってしまいます。この問題については、現在議論が進められています。

## 付録 2『キーボードショートカット』

ここでは、現在の Windows 環境で幅広く使用されているショートカットキーと、関連付けられている機能の一部を紹介します。

もし、制作中のアプリケーションに、ここで挙げる機能に該当するものがあれば、同一のショートカットキーを割り当てることを推奨します。また逆に、異なる機能を割り当てているのならば、別のショートカットキーに変更して下さい。

なお、Microsoft 社が公開している技術文書『キーボード ユーザー インターフェイス設計のガイドライン』には、より多くのショートカットキーが紹介されています。

**（機能名）**
　（操作キー）

**決定**
　Enter

**キャンセル**
　Escape

**項目の選択・選択解除**
　Space

**メニューバーへのアクセス**
　Alt

**最小化などのシステムメニュー表示**
　Alt+Space

**コンテキストメニューを表示**
　アプリケーション、または Shift+F10

**カーソルの上下左右方向への移動**
　上下左右矢印キー

**階層構造になっているリストの次階層の展開**
　右矢印キー

**階層構造になっているリストの現在階層を閉じる、前の階層に戻る**
　左矢印キー

**ダイアログ・ボタン間の移動**
　Tab、Shift+Tab

**ウィンドウ内のタブの切り替え**
　Ctrl+Tab、Ctrl+Shift+Tab

**AccessKey の指定されたダイアログ・ボタンを選択**
　Alt+(アクセスキー)

**文頭へ移動（一行の場合は行頭へ移動）**

Ctrl+Home

**文末へ移動（一行の場合は行末へ移動）**

Ctrl+End

**一画面分、または複数回分上のオブジェクトへ移動**

PageUp

**一画面分、または複数回分下のオブジェクトへ移動**

PageDown

**前後の段落へ移動**

Ctrl+上下矢印キー

**段落の行頭・リスト項目の先頭位置へ移動**

Home

**段落の行末・リスト項目の末尾へ移動**

End

**前後の文節（句読点・スペース等の位置）へ移動**

Ctrl+左右矢印キー

**全ての項目を選択**

Ctrl+A

**指定した範囲の項目を選択**

Shift+上下左右矢印キー

**文字列やオブジェクトを検索**

F3、または Ctrl+F

**文字列を置換**

Ctrl+H

**文字列やオブジェクトの切り取り（移動するオブジェクトの選択）**

Ctrl+X

**文字列やオブジェクトの複写**

Ctrl+C

**文字列やオブジェクトの貼り付け**

Ctrl+V

**直前に操作したオブジェクトを操作前の状態に戻す**

Ctrl+Z

**操作前の状態に戻した変更を再度反映**

Ctrl+Y

**画面の情報を最新のものに更新**

F5

**新しいファイルを作成**

Ctrl+N

**既存のファイルの読み込み**

Ctrl+O

**既存のファイルを印刷**

Ctrl+P

**編集中のファイルを上書きで保存**

Ctrl+S

**編集中のファイルを別名で保存**

Ctrl+Shift+S

**ヘルプを表示**

F1

**アクティブなアプリケーションを終了**

Alt+F4

**スタートメニューを開く**

Windows、またはCtrl+Escape

**タスクバーを選択**

Windows+B

**エクスプローラの表示**

Windows+E

**デスクトップ画面を表示**

Windows+M

**ファイル名を指定して実行**

Windows+R

**ウィンドウの選択**

Windows+Tab

**音の再生・一時停止**

Ctrl+P、またはSpace

**音の停止**

Ctrl+S、またはCtrl+Period

## 付録３『用語集』

**（用語名）**

（用語の説明）

**AccessKey**

特定のキーを入力すると項目が選択できる仕組み。携帯電話向けホームページ制作で多用されるが、キーボードショートカットキーにも使える。

**API(Application Program Interface)**

アプリケーションプログラミングのための命令群。メッセージを表示する、音声で読み上げる、といった命令が集められている。

**AUI(Auditory User Interface)**

ユーザインタフェースの一つ。聞こえてきた音を頼りに操作を進める。

**DLL(Dynamic Link Library)**

アプリケーションで利用する機能や命令を集合したファイル。多くのWindows 向けアプリケーションはこの DLL へもアクセスしている。

**GUI(Graphical User Interface)**

ユーザインタフェースの一つ。画面上のアイコンやボタンを選択することで操作を進める。

**HTML ヘルプ**

ヘルプ形式の一つ。通常の HTML と違う点は、複数の WEB ページを一つのファイルに統合していること、および目次が自動的に付与されていることである。

**MP3(MPEG audio layer 3)**

音声圧縮フォーマットの一つ。通常の音声データを 10 分の 1 程度に圧縮できる。本書のサンプル音声にも使用している。

**MS-DOS**

オペレーティングシステムの一種。Microsoft 社が Windows 以前に発表していた。

**PDF(Adobe Portable Document Format)**

アドビ システムズ社から提唱されたドキュメントのフォーマット。ISO32000 に登録されている、現在主流な電子文書の配布形式。

**SAPI(Speach API)**

API の一つ。音声合成による文字列の音声化機能を提供する

**Wave**

音声フォーマットの一つ。データ量が多いため、通常は MP3 等に圧縮する。

**アイコン**

コンピュータ上で、ファイルや特定の操作コマンドなどを絵に置き変えたもの。ユーザがその絵から、状況やその後の操作を想起できるようなデザインになっている。

**アクセシビリティ**

アクセス可能性。製品の全ての機能にアクセスし利用できるかどうか。

**アクセス**

指定した機能を利用するためにユーザが行なう一連の操作のこと

**アクティブ**

ユーザから見た現在の作業の対象。

**アプリケーション・アプリケーションソフトウェア**

ドキュメント作成や計算、WEB ページの閲覧などの、特定の目的を達成するために利用するソフトウェアの総称。

**イベント**

何らかの出来事。アプリケーション開発においては、例えばユーザがキーを押した、一定の時刻になった、などの状況の変化を指す。

**ウィンドウ内のタブ**

ウィンドウをいくつかのフレームに分割している状態での個々のフレームの意。

**エクスプローラ(Explorer)**

既存のファイルやディレクトリを表示し、管理・編集するためのWindows アプリケーション。

**オープンソース**

ソフトウェア開発形態の一つ。ソースコードを公表することで、複数人の開発者が共同で開発を行なう。

**カーソル**

ユーザが特定の項目を選択するために、マウスやキーボードを用いて移動させるポインター(矢印など)。

**キーボード**

本書では入力デバイスの一つ。端末に対して文字情報を入力するために用いる。

**クリップボード**

文字列や画像などを一時的に保管しておくための領域。

**コンテキストメニュー**

右クリックメニュー・ショートカットメニューなどとも呼ばれる。アプリケーションキーを押すことによって表示され、通常は実行可能な項目が並んでいる。

**システム標準**

本書では主に Windows アプリケーションにおいて使われている形式。

**ショートカット**

特定の機能を、本来の手順よりも短い操作で実行できる仕組み。

**スクリーンリーダー**

障害者支援ソフトウェアの一つ。画面に表示された情報やユーザが操作した結果を音声や点字でガイドする。

**ステレオ**

本書では音声出力方式の一つ。左側・右側から異なる音を鳴らした際に成立する。

**ソフトウェア**

コンピュータに一定の作業を行わせるために記述された命令群。

**ダイアログ**

文字の入出力、項目の選択などに用いる画面。ダイアログボックスとも呼ばれる。

**タイムラグ**

本書では、ユーザが指示をしてから作業が実行されるまでの間の時間を指す。

**タスクバー**

現在起動しているアプリケーションのアイコンを表示するための領域。常駐するアプリケーションや時計などが表示されることが多い。

**ツールバー**

現在起動しているアプリケーションが最小化された時に格納される領域。個々の画面を選択すれば、元のウィンドウが表示される。

**ディレクトリ・フォルダ**

何らかの規準によって整理したファイルを入れておく箱のようなもの。

**デコード**

MP3 や JPEG 等の圧縮データを元のデータに戻す処理。圧縮データはそのままでは表示・再生できないためこの処理が必要である。

**デスクトップ**

アプリケーションが一つも起動していない時に表示されるウィンドウ。この画面には各アプリケーションを実行するためのショートカットアイコンが配置されており、ユーザがそれを選択し実行する。

**フォーカス**

ユーザが、カーソル等を移動させたことによって、特定の項目を選択すること。

**フリーズ**

本書では無限ループと同意。ソフトウェアが操作を受け付けず、且つ画面の変化やフィードバックが一切無い状況を指す。

**プルダウン**

表示方法の一つ。通常は一つの項目が表示されている部分があり、選択することで残りの項目が表示される。

**プルダウンメニュー**

クリックすることで項目がプルダウンしていくメニュー。現在の Windows で主流のユーザインタフェース。

**ポインティングデバイス**

ディスプレイ上に存在する矢印(ポインタ)を操作することでパソコンを制御する機器。マウスやトラックボールの総称。

**ホームポジション**

キーボードの「F」・「J」キーを中心とした指の配置。このキーに人差し指を置き、「A」・「S」・「D」、「K」・「L」・「SEMICORON」キーに残りの指を置く。

**ボタン**

コンピュータ上に存在する仮想のボタン。ユーザがそれを押すことによって、ボタンに設定された内容がアプリケーションに送信される。

**ポップアップ**

ある操作によって、それまで存在しなかった画面などが現れること。

**マイ・コンピュータ**

ハードディスクや CD ドライブなどの内容を参照するためのフォルダ。「マイ・ドキュメント」はここからも参照できる。

**マイ・ドキュメント**

個々のユーザが作った資料などのファイルを整理するために Windows で定義されたフォルダ。実際には音楽やビデオなどのファイルも入れられる。

**メニューバー**

ウィンドウとタイトルバーの間にあるスペース。スペース内には、そのアプリケーションを利用するために必要な機能の一覧が並べられる。

**ユーザビリティ**

既存の製品を、指定したユーザが特定の環境下で、一定の手順で利用した時の、その利用目的の達成度を計る指標。

**ラベル**

ウィンドウやボタンに設定することのできる名札のようなもの。

**リアルタイム**

即時的・同時的なこと。ソフトウェアでは、時間の経過に合わせて、自動的に情報を変更していく様を表す。

**リストボックス**

ダイアログボックスの一つ。複数の項目を参照したり、任意の項目を選択するために利用する。

**音声合成エンジン**

音声合成に必要な波形データの集合体。文字データを取り込むことで、読み上げに当たる波形データを出力できる。

**詳細読み**

テキストデータの読み上げ方式の一つ。選択している一文字に対して、文字の内容や書体などを読み上げる。

**全文読み**

テキストデータの読み上げ方式の一つ。文書を文頭、または選択した位置から文末までを連続して読み続ける。

**標準API**

本書では、Windowsが登場した頃から使われているアプリケーションユーザインタフェースのこと。

**無限ループ**

何らかの原因でソフトウェアが同一の処理を永久的に繰り返し、それを停止させる手段が無い状態。

**論理的に構成する**

本書では、あるデータをソフトウェアなどが完全に解釈できるようにすること。

## 付録４『HTML ヘルプのキーボード操作』

本付録は、第７章『ドキュメント』の節で紹介した HTML ヘルプについて、どのような画面構成をしているのか、そしてキーボードでどのように操作するのかについて紹介しています。

### ◆HTML ヘルプの画面構成

HTML ヘルプは、左右二つの画面に分かれています。左側の画面には、目次、キーワード検索、そして文字列検索の画面があります。右の画面には、ヘルプの本文が記述されています。

目次の画面は、ツリービュー形式のリストボックスになっています。上下の矢印キーを操作して項目を選択し、エンターキーを押すと、右の画面にその説明が表示されます。

また目次の中には、右矢印キーを押すと、その章に属する節に当たる項目が表示されることがあります。逆に、左矢印キーを押すと、節の項目は見えなくなります。

キーワード検索および、文字列検索の画面は、入力した文字が含まれるヘルプ記事を表示します。キーワードは、開発者が予め準備してあるテキストから選択することができます。

キーワード検索画面には、検索のための文字列の入力ボックス、キーワードの候補を選ぶためのダイアログボックス、そして確定し表示するボタンがあります。それぞれの切り替えにはタブキーを使います。

文字列検索画面には、検索のための文字列の入力ボックス、検索開始ボタン、そして検索結果を表示するリストボックスがあります。キーワード検索と同じように、タブキーで切り替えることができます。

### ◆操作方法

目次・キーワード検索・文字列検索の画面を行き来するには、Ctrl+Tab キーを押します。この操作は、目次・キーワード検索・文字列検索という３つのフレームがあり、それを切り替えていくことに似ています。

また、目次へは「Ctrl+C」、キーワード検索へは「Ctrl+N」、文字列検索へは「Ctrl+S」キーを押すことで直接アクセスすることもできます。

ヘルプの記事を読むには、F6 キーを押します。もう一度 F6 キーを押すと、目次などの画面に戻ることができます。

ヘルプの記事は、インターネットを読む時と同じ操作で読むことができます。記事の中のリンクをクリックすれば、リンク先の記事を読むこともできます。

ただし、PC-Talkerを利用している場合、ダイレクトキーの利用有無に関わらず、ヘルプ記事内では行読み・文字読みができません。Ctrl+Alt+Aキーで全文を読むか、Ctrl+Sで音声コピーを実行し、テキストエディタなどに貼り付けて読んで下さい。

## 付録５『スクリーンリーダーで読み取れる情報』

本付録は、スクリーンリーダー『NVDA』で本書の序章を読み上げた内容をそのまま掲載しています。

※NVDA には、音声で実際に読み上げた内容を画面上に出力する機能が搭載されています。NVDA を利用することで、誰でもほぼ同等の出力結果を確認することができます。

見出し　レベル１
ＭＳ ゴシック　16pt　太字
序章
見出し　レベル２
14pt
◆はじめに
ＭＳ 明朝　12pt　太字なし
現在、私達の生活を支えているパソコンには、さまざまなアプリケーションソフトウェアが搭載されています。その種類は豊富で、ワープロ・表計算・電子メールといった現代の必需品となっているものから、音楽・アート・ゲームといった娯楽、さらには日々のスケジュールの管理や、会計・経理を行なうアプリケーションまで多岐に渡ります。
しかし、全てのユーザがその恩恵を受けられているわけではありません。例えば視覚障害があり画面の表示が見えないユーザは、現代のグラフィック中心のアプリケーションソフトウェアをほとんど利用することはできません。
著者もこの視覚障害ユーザーとなった一人です。視力を失ったことで、それまで自然に行なえたはずのパソコン操作の全てが不可能になりました。
空行
しかし、視覚障害ユーザーでもそれらのアプリケーションソフトウェアが使えるようになる方法があります。それは、操作するために必要となる情報を、音声や効果音を用いてフィードバックすることです。
このユーザインタフェース
太字
AUI(Auditory User Interface)
太字なし
は、著者を初めとする多くの視覚障害ユーザーに普及しました。少なくともその結果、著者は音の情報を頼りに本書を執筆できたのです。
空行
とはいえ、まだ視覚障害ユーザーが全てのアプリケーションソフトウェアを使うことはできません。現代の GUI(Graphical User Interface)ユーザーを規準に制作したソフトウェアの全てを音声や効果音で表現することは完全ではありません。

また、視覚障害ユーザーと、音声や効果音を頼りに操作する手法、考え方も、まだ確立されているわけではありません。そこで本書では、この視覚障害ユーザーと、ユーザインタフェースにスポットライトを当てることにしました。
空行
見出し　レベル2
ＭＳ ゴシック　14pt　太字
◆AUI
ＭＳ 明朝　12pt
AUI
太字なし
とは、コンピュータを操作するためのユーザインタフェースの一つです。
AUIの特長は、ユーザがマウスやキーボードを用いて行なった操作の結果や、現在のパソコンがどんな状態であるかなどを、音声や効果音でフィードバックすることです。
空行
このユーザインタフェースを最も簡単に実現できるツールは、
太字
スクリーンリーダー
太字なし
と呼ばれるアプリケーションソフトウェアです。
スクリーンリーダーとは、現在多くの視覚障害のユーザに利用されている支援ソフトウェアです。画面に表示された情報や、ユーザが操作した結果を音声で読み上げることができます。
空行
見出し　レベル2
ＭＳ ゴシック　14pt　太字
◆ガイドラインの概要
ＭＳ 明朝　12pt　太字なし
本書は、AUIを利用するユーザが使いやすいソフトウェアを開発するためのガイドラインです。
ユーザビリティとは、既存の製品やサービスを、ある限定された条件下で使った時、その利用目的に対する達成度合いを総称したものです。より詳しい定義については、
リンク
ISO 9241-210『Ergonomics of human-system interaction -- Part 210: Human-centred design for interactive systems』
(日本版
リンク
JIS Z8530『人間工学—インタラクティブシステムの人間中心設計プロセス』
)を参照して下さい。

このガイドラインで目標としていることは、AUI ユーザーが、正確に作業を遂行できること、作業時間を短縮できること、そして、利用する過程で感じるストレスを軽減できることを念頭に置いたアプリケーションソフトウェアを開発することです。
空行
また同時に、本書は AUI ユーザーへ向けての
太字
アクセシビリティ
太字なし
に配慮してソフトウェアを設計するためのガイドラインでもあります。
太字
アクセシビリティ
太字なし
とは、「ユーザが指定した機能にアクセスできるようになっているかどうか」を意味する言葉で、前述の
太字
ユーザビリティ
太字なし
の
太字
正確に遂行できること
太字なし
が達成されるために必須となる要素です。
本書では主に第２部で、AUI ユーザーが全ての機能にアクセスし操作できるアプリケーションソフトウェアを開発するための要点を記述しています。
空行
見出し　レベル 2
ＭＳ　ゴシック　14pt　太字
◆ガイドラインの特長
ＭＳ　明朝　12pt　太字なし
このガイドラインには、次のような特長があります。
リスト 3 項目
・AUI ユーザーを
太字
知る
太字なし
セクションを設けているので、AUI ユーザーにとってわかりやすいソフトウェアが制作できます。

・音声や効果音を頼りに情報を素早く検索できるための用法が盛り込まれているので、目的の作業を素早く遂行できるソフトウェアが制作できます。
・操作ミスや混乱を防ぐと共に、誤った際の対処法についても紹介しているので、アプリケーションを使う上でのストレスを軽減できるようなソフトウェアが制作できます。
リスト終了
空行
またこのソフトウェアは、AUI ユーザー以外のユーザにとっても、以下のように使いやすくなる特長があります。
リスト 3 項目
・画面を見続けなくても使用できるソフトウェアになるので、注視することによる疲労を少なくできます。
・全ての機能をキーボードでも操作できるようになるので、マウス操作の苦手なユーザや、ショートカットキーを駆使して素早く操作したいユーザのニーズにも対応できます。
・ロービジョン(弱視)なために画面を拡大しているユーザや、小さなディスプレイでパソコンを利用しているユーザにとって、通常の GUI アプリケーションよりも使いやすくなります。
リスト終了
空行
本書は、他のユーザビリティガイドラインと比較すると、以下のような特長を持っています。
リスト 2 項目
・障害当事者自身が制作したガイドラインなので、視覚障害ユーザーの視点を盛り込んでいます。
・著者は 5 年以上に渡りソフトウェア制作の経験を積んでおり、現在もアプリケーション制作を行なっています。従って、制作に実装できる用法も説明に加えています。
リスト終了
空行
見出し　レベル 2
ＭＳ ゴシック　14pt　太字
◆本書の構成
ＭＳ 明朝　12pt　太字なし
本書は、3 つの部で構成しています。
リンク　太字
第１部 AUI ユーザーを学ぶ
太字なし

AUI を利用している視覚障害のユーザの特長や課題、AUI とはどういった特長があるのかを紹介します。
この部には、具体的なソフトウェア開発に活用できるガイドラインはありません。しかし、AUI ユーザーを知ることで、第２部以降のガイドラインがわかりやすくなりますので、ぜひ目を通して下さい。
リンク　太字
第２部 AUI アクセシビリティを学ぶ
太字なし
AUI ユーザビリティの基本として、
太字
アプリケーションの全ての操作を AUI で実現できる
太字なし
ことを念頭に置いた部です。そのために、アプリケーション開発時に配慮して欲しい要点を紹介しています。
この部で挙げている多くの記述は、ユーザビリティの仲でも、
太字
全ての機能にアクセスできる
太字なし
という意味を持つ用語
太字
アクセシビリティ
太字なし
に関する内容です。
リンク　太字
第３部 より深く AUI ユーザビリティを学ぶ
太字なし
AUI の中核である音声情報の取り扱い方について紹介します。第２部のガイドラインを理解した上で、より優れたユーザビリティを追求したい時に活用して下さい。
この部では、どのように音声情報を提示すれば AUI ユーザーに使いやすいアプリケーションになるのか、その要点を中心に記述しています。
空行
見出し　レベル 2
ＭＳ ゴシック　14pt　太字
◆対象とする読者
ＭＳ 明朝　12pt　太字なし
本ガイドラインは、次のような方には特に読んでいただけることを望みます。もちろん、当てはまらない方でも、ぜひ読んでいただきたいと考えています。
リスト 3 項目

・アプリケーション開発やコンテンツ制作に携わっている方
・障害者や障害者支援に関心のある方
・音や音声のユーザインタフェースに関心のある方
リスト終了
空行
ただし、本書には以下の内容については充分には記述していません。
リスト 3 項目
・音声入力や、点字・触覚に関わるユーザインタフェース
・具体的なソースコードの書き方や、ソフトウェアの名称
・Windows 向けアプリケーションソフトウェア以外の製品へのアクセシビリティの応用
リスト終了
10pt
※本書を Windows 向けアプリケーションに限定して記述している理由は、日本では Windows 向けスクリーンリーダーの入手が最も容易であるからです。

## 著作情報

**著者：**

諸熊 浩人(モロクマ ヒロト)

株式会社 U'eyes design ユーザセンタードデザイン事業部

E-MAIL：morokuma_hiroto@ueyesdesign.co.jp

**発行：**

*株式会社 U'eyes design(ユー・アイズ・デザイン)*

http://www.ueyesdesign.co.jp/

郵便番号：224-0001

所在地：横浜市都筑区中川 1 丁目 4-1 ハウスクエア横浜 4 階

Tel：045-914-7820

FAX：045-914-7822